

Responder Identification in Clinical Trials

Dissertation

zur Erlangung des Grades Doktor der Naturwissenschaften (Dr. rer. nat.)

an der Fakultät für Mathematik, Informatik und Statistik

der Ludwig-Maximilians-Universität München

vorgelegt von

Victoria Kehl

am 28. Juni 2002

Referent: Prof. Dr. K. Ulm

Koreferent: Prof. Dr. L. Fahrmeir

Rigorosum: 21. November 2002

Acknowledgements

This research was done during my work at the Institute for Medical Statistics and Epidemiology (IMSE) at the Technical University – Munich. It was supported by grant UL 94/11-1 of the German Research Community (DFG) from December, 1999 until December, 2000 and by DFG's Special Research Areas (SFB) 386 "Statistical Analysis of Discrete Structures," project B7: "Prognoses for Cardiac Arrhythmia Patients" since January 2001.

I would like to thank my advisors Prof. Dr. Ludwig Fahrmeir and Prof. Dr. Kurt Ulm for their involvement in SFB, which assured perfect research atmosphere with sufficient funding for computer, software, literature, conferences, and other events that made contact with other researchers possible on national and international level. I would like to thank Prof. Dr. Ulm especially for all his support throughout the research period, for his friendliness and patience, for believing in me, for the countless discussions and useful suggestions, and for his ever readiness to share knowledge and networking.

Many thanks also to the cardiologists Dr. Petra Barthel and Prof. Dr. Georg Schmidt for providing the EMIAT data and for clearing the way through the jungle of medical terminology, as well as for their input on the medical feasibility of the EMIAT models. Thank you, I consider myself lucky to have worked with such a friendly and knowledgeable team.

Further, I would like to thank all my colleagues at IMSE for their moral support and for providing a friendly atmosphere at work, which made the way through the almost daily ups and downs of my research bearable.

Last, but not least, I am grateful to my family and friends who, unlike me, never doubted that this dissertation would become reality.

Thank you all!

CONTENTS

1	Introduction	1
1.1	Aims	1
1.2	Outline	1
2	Responders and non-responders	3
2.1	Motivation	3
2.2	Definitions	5
3.3	Assumptions	6
3	Preliminary: Residuals to the Cox-PH model	7
3.1	The Cox-PH model	7
3.2	Schoenfeld residuals	11
3.3	Martingale residuals	12
3.4	Score residuals	15
3.5	Deviance residuals	17
3.6	Log-odds and normal deviate residuals	20
3.7	Suitable residuals for responder identification	21
4	The classical approach: Cox-PH model with interactions	28
4.1	Definition	28
4.2	Responder identification	29
5	Classification and regression trees (CART) – recursive partitioning	30
5.1	Growing a regression tree	34
5.1.1	Splitting	34
5.1.2	Stopping criteria	36
5.1.3	Pruning	36
5.2	Tree Performance	37
5.3	Responder identification with regression trees	38
6	PRIM – Patient rule induction method (Bump Hunting)	39
6.1	General structure of the PRIM model	40
6.2	Box construction	42

6.3	Responder identification with bump hunting	46
7	Stabilization of Bump Hunting	48
7.1	Stabilizing with bootstrapping	48
7.2	Algorithm	50
7.3	Discussion	52
8	Identification of responders and non-responders	53
8.1	Algorithm using Bump Hunting	53
8.2	A note on survival curve difference	55
8.3	Changes to the responder identification algorithm if CART is used	58
8.4	Covariate considerations	59
9	Simulation study	60
9.1	Methods	60
9.2	Results	65
9.2.1	Cox-PH with interaction	65
9.2.2	Regression trees	69
9.2.3	Bump Hunting	72
9.3	Comparison	76
9.4	Implementation	77
10	Applications: EMIAT	78
10.1	Data	78
10.2	Previous investigations	78
10.3	Cox-PH with interaction	81
10.4	Responder identification with CART	84
10.4.1	The prognostic model	84
10.4.2	The predictive model with continuous factors	85
10.4.3	The predictive model with categorized factor	90
10.5	Responder identification with Bump Hunting	97
10.6	Comparison and discussion	103

Conclusions	109
Summary	109
Outlook	110
 Appendix A: Proofs	 112
Appendix B: Simulation study plots	117
Appendix C: Algorithmic Implementation	122
Bibliography	142

1. INTRODUCTION

Some of the main areas of medical research are prevention and cure of pathological conditions and increase of life quality (and quantity). This is directly linked to the development of new treatments, which work faster, are more effective and with less side effects than the old ones. The term *treatment* here is used in a broad sense to cover the most common chemical (i.e. medication coming from the pharmaceutical industry), as well as surgical, mechanical, radial, and psychological treatment. In order to judge the efficacy of a new treatment objectively, a clinical trial needs to be designed and evaluated. This is one of the major working fields of biostatistics as well.

Just as any other modern science, biostatistics is a hybrid science. It is mainly based in the area of statistics, but it reaches over to medicine, mathematics, and computer science as well. This research represents such a mixture of statistics, optimization, and computer science in the search for improvements in the clinical trial evaluation process.

1.1 Aims

The subject of this thesis is responder analysis. The term *response* up to now appears only in clinical trials, in which surrogate markers are used to describe the effect of the treatment when that effect is other than to prevent an event.

Example 1: In oncology, the desired effect of a treatment may be reduction of tumor size, whereas the outcome of interest (called event) may be death. Then a *responder* is a patient who experienced tumor reduction or complete remission and a *non-responder* is a patient who's tumor did not change or grew. Notice, that does not necessarily mean that responders lived longer.

Example 2: If a headache medicine is tested, the event may not be defined to be death, but recurrence of headache. There is no existing definition of responder in this

case, but, in general, the goal is to prolong the event-free (headache-free) period. Notice, side effects and death due to the drug should also be tested before the drug is approved, but this would not be the main aim of the study.

The European Myocardial Infarction Amiodarone Trial (EMIAT), described in chapter 10 falls in the case described by the second example. Amiodarone is an anti-arrhythmic drug. In order to approve the drug, an anti-arrhythmic effect as well as prolonging of life had to be shown, so event was defined to be all cause mortality. Once again, no definition of responder is available in this case, but the final goal was to show, that Amiodarone increases the event-free period (prolongs life).

The classical definition of *responder* is altered in this research in order to fit the more general clinical trial situation, in which the wished effect of a treatment is increasing the event-free period of the treated patient (example 2).

1.2 Outline

The term responder is redefined in chapter 2 and distinction between prognostic and predictive factors is made. The rest of this thesis focuses on methods for identification of responders. Chapter 3 gives an overview of residuals to the Cox-PH model, which can be used as a prognostic model. The ability of different residuals to identify predictive factors is analyzed. The classical approach for responder identification is presented in chapter 4. Chapter 5 gives an overview of recursive partitioning while focusing on regression trees. Bump hunting is presented in chapter 6. Both regression trees and bump hunting can be used for responder identification purposes. An attempt to stabilize the bump hunting algorithm through bootstrapping is given in chapter 7. Chapter 8 lists the steps of the proposed responder identification algorithm, as well as some general suggestions on its use. The results of a simulation study which compares all discussed versions of the responder identification algorithm are presented in chapter 9. Finally, the last chapter presents the results of the responder analysis performed on the EMIAT data set.

2. RESPONDERS AND NON-RESPONDERS

2.1 Motivation

Consider the common clinical trial situation in which the ability of a new treatment to prevent an event is tested. Patients are randomized into two groups: one receiving the classical treatment (or placebo) and the other receiving the new treatment. Not rarely, the outcome of such trials shows no difference in the survival probabilities of the two treatment groups (see figure 2.1). But still, it could happen that certain subgroups of patients show improved survival under the new treatment, while others appear to suffer from it (see figure 2.2).

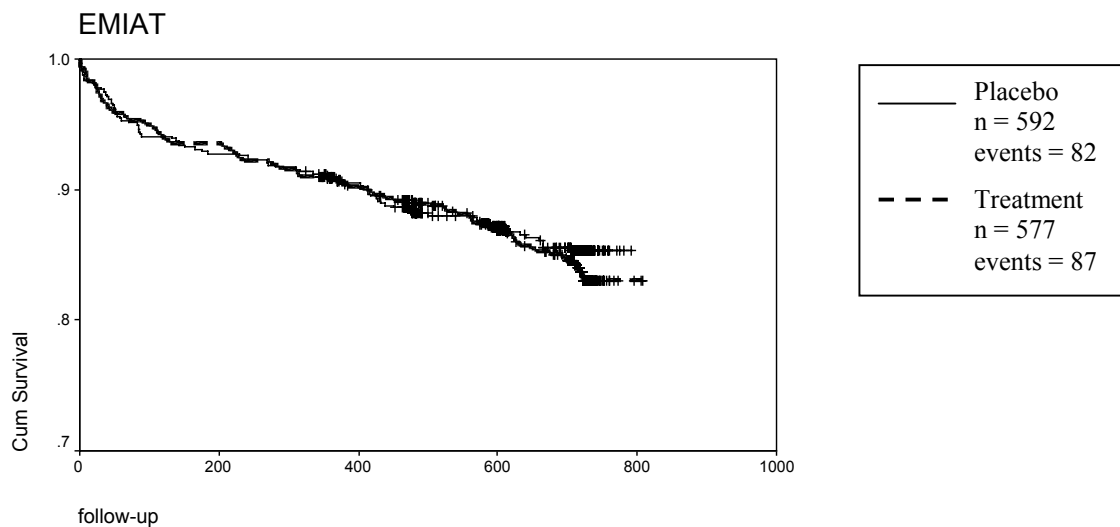


Figure 2.1: *Kaplan-Meier survival curve estimates for the placebo and Amiodarone treatment arms of the EMIAT study (details on the study in chapter 10).*

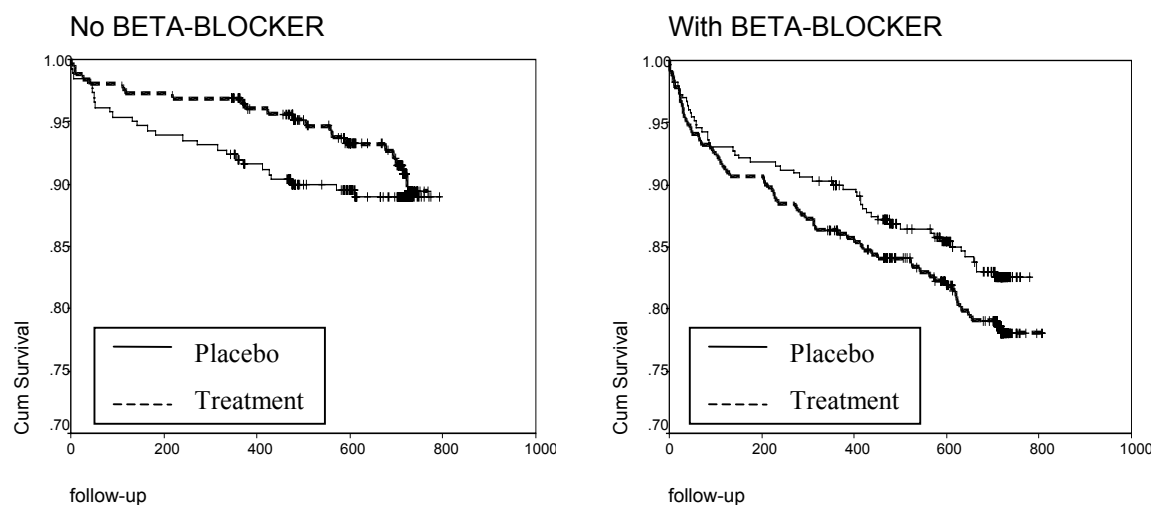


Figure 2.2: *Comparison of the Kaplan-Meier survival curve estimates for the placebo and Amiodarone arms of EMIAT for the subgroups of patients on and off beta-blocker.*

Suppose the survival time of a patient in the new treatment group is greater than the overall survival time. There can be three reasons for this phenomena:

1. Chance: we cannot predict or account for occurrence by chance in any way or form in a model.
2. The patient has a prognosis better than the average, due to the specific **prognostic factors** that he enjoys.

For example, if younger patients in general have different prognosis than older ones, independently of their treatment group, then we would say that age is a **prognostic factor**.

We can account for prognostic factors, provided that they have been measured, by developing a prognostic model on the classical treatment (or placebo) group. Such model can be the Cox proportional hazards model, a survival regression tree, or even the more exotic neural network – it simply has to be a model predicting

survival (or hazard) in the classical treatment arm of the study. Factors found to be significant in such a model are called prognostic.

Notice, the so defined factors would be prognostic in the real sense of the term only if they are found on a placebo arm. If the new treatment is tested against a classical treatment, the factors would be "prognostic" only with respect to the new treatment and not in general. To avoid confusion, for the rest of this thesis we will call both factor types prognostic.

3. The new therapy is really working. The purpose of this research is to explore methods of identifying patients with special reactions to the new treatment (those could be positive as well as negative), which are different from the whole patient population and cannot be explained by prognostic factors. In such cases **predictive factors** are responsible for the difference in survival.

"Predictive factors:

- *Any factor which predicts how a patient will do with adjuvant systemic therapy*
- *Looks for differential effect of treatment*
- *To understand predictive factors subgroup analyses are required..."*

Silva & Zurrida, 2000

Note, that a factor can have both prognostic and predictive power, if its prognostic value is different in the two treatment groups. If only one predictive factor is involved (or several independent predictive factors), it can be found by adding an interaction term involving the treatment randomization index and the predictive factor in question in a model which already accounts for prognostic factors (see chapter IV). Methods for identifying groups of predictive factors are described in chapters V and VI.

2.2 Definitions

We define **positive responders** to be patients under the new treatment, who benefit from it. Their benefit is manifested in the fact that their survival time is longer than that of patients with the same characteristics (predictive factors), randomized in the classical treatment group.

We define **negative responders** to be patients under the new treatment who are harmed by it. Their survival time is shorter than that of a similar, described by predictive factors, group of patients under the classical treatment.

Consequently, **non-responders** would be patients who are neither positive nor negative responders. Their survival time does not differ from similar patients under the classical treatment.

We are interested in identifying responders – both positive and negative.

2.3 Assumptions

Responders are identified and characterized by predictive factors. For the successful identification of predictive factors we need the assumption that all prognostic factors are already correctly accounted for in a prognostic model. This is a strong, but not unreasonable assumption.

If this assumption is not fulfilled, we run the risk of wrong conclusions. For example, we may conclude a therapy effect where there is none. The patient just has a better prognosis to begin with, which was not recognized by the prognostic model. The opposite can also be falsely concluded. We can conclude that a patient is harmed by the new treatment, when in fact he/she does not react to the new treatment any differently than the rest of the group. The patient just has a worse prognosis due to a prognostic factor which was not yet accounted for.

3. PRELIMINARY: RESIDUALS TO THE COX-PH MODEL

Before describing methods for responder identification, we need a prognostic model. One possible method of constructing a prognostic model is the Cox proportional hazards model (Cox-PH), which is well known and widely used in the area of survival analysis (Cox, 1972). This chapter gives some preliminary knowledge of the Cox-PH model and its residuals, as well as the foundation for their possible use for responder identification purposes.

3.1 The Cox-PH model

The Cox-PH model describes a population of n patients with follow-up times t_i , final status δ_i , and a set of K covariates $x_i = (x_{1i}, x_{2i}, \dots, x_{Ki})$ by describing the hazard rate for each patient i from 1 to n as:

$$\lambda(t, x_i) = \lambda_0(t) \cdot e^{\beta' \cdot x_i} \quad (3.1)$$

The effect of the covariates is assumed to be log-linear and independent of time. Proportionality of hazards is also assumed, i.e. the failure rates of any two individuals are proportional, which means that their hazard ratio is constant over time. Naturally, one can use extensions to the Cox model as prognostic models as well. For simplicity, we will restrict this research to the classical Cox-PH model.

Estimating β

As described by Cox & Oakes (1984) and Marubini & Valsecchi (1995), one needs to use partial likelihood for estimation of the coefficient vector β , since the baseline hazard $\lambda_0(t)$ in the Cox-PH model is not specified parametrically. Suppose a total of J events

occur in a sample of N subjects. Let $t_{(j)}$ denote the ordered failure times, $j = 1, \dots, J$. Let $R(t)$ be the set of subjects at risk at time t and R_j be the set of subjects at risk at time $t_{(j)}$ (i.e. $R_j = R(t_{(j)})$). Let x_j be the vector of K covariates for the subject who fails at time $t_{(j)}$ and x_i be the vector of covariates for the i^{th} subject, $i = 1, \dots, N$. Assuming that only one individual fails at $t_{(j)} \in (t, t + \Delta t)$, the probability that it is an individual with covariates x_j is (Marubini & Valsecchi, 1995):

$$\frac{\lambda(t_{(j)}, x_j)}{\sum_{i \in R_j} \lambda(t_{(j)}, x_i)}$$

Then the function describing the entire failure pattern for the set of J deaths is the product (Cox & Oakes, 1984):

$$L(\lambda_0(t), \beta) = \prod_{j=1}^J \frac{\lambda(t_{(j)}, x_j)}{\sum_{i \in R_j} \lambda(t_{(j)}, x_i)}$$

Given the Cox-PH model (3.1), the likelihood function simplifies to:

$$L(\beta) = \prod_{j=1}^J \frac{e^{\beta' \cdot x_j}}{\sum_{i \in R_j} e^{\beta' \cdot x_i}},$$

which is a partial likelihood function depending only on the unknown β values (and the known x values). The unknown values of β are then estimated by the values $\hat{\beta}$, which maximize the partial log-likelihood:

$$LL(\beta) = \sum_{j=1}^J \left[\beta' \cdot x_j - \log \left(\sum_{i \in R_j} e^{\beta' \cdot x_i} \right) \right] = \sum_{j=1}^J l_j,$$

where l_j is the contribution to the log-likelihood for failure time $t_{(j)}$ (Marubini & Valsecchi, 1995). The estimates $\hat{\beta}$ of β are found by equating to zero the K first partial

derivatives of $LL(\beta)$ with respect to β_k , $k = 1, \dots, K$, and solving the system of K equations, each of which has the form:

$$U_k(\beta) = \frac{\partial LL(\beta)}{\partial \beta_k} = \sum_{j=1}^J \frac{\partial l_j}{\partial \beta_k} = 0 \quad (3.2)$$

Concentrating just on the derivative contribution to the sum:

$$\frac{\partial l_j}{\partial \beta_k} = x_{kj} - \frac{\sum_{i \in R_j} x_{ki} \cdot e^{\beta' \cdot x_i}}{\sum_{i \in R_j} e^{\beta' \cdot x_i}} \quad (3.3)$$

it can be generalized to:

$$\frac{dl_j}{d\beta} = x_j - \frac{\sum_{i \in R_j} x_i \cdot e^{\beta' \cdot x_i}}{\sum_{i \in R_j} e^{\beta' \cdot x_i}} \quad (3.4)$$

for the entire vector of K covariates for a subject failing at time $t_{(j)}$.

If more than one deaths occurred at time $t_{(j)}$, the partial log-likelihood can be modified to:

$$LL(\beta) = \sum_{j=1}^J \left[\beta' \cdot s_j - d_j \cdot \log \left(\sum_{i \in R_j} e^{\beta' \cdot x_i} \right) \right] = \sum_{j=1}^J l_j ,$$

where s_j is the sum of all covariate vectors of the subjects who fail at time $t_{(j)}$ and d_j is the number of such subjects. This approach was proposed by Peto (1972).

Estimating the baseline hazard:

A simple maximum likelihood estimator of $\lambda_0(t)$ was proposed by Breslow (1974), which is now widely used for baseline hazard estimation. To deal with the censoring problem, Breslow assumed that the hazard is constant between two consecutive failure times. The baseline hazard was estimated separately in each of the intervals between failure times: $(t_{(j-1)}, t_{(j)}]$, where $j = 1, \dots, J$. Assumed are: $t_0 = 0$ and censoring within the interval occurred at the beginning of the interval, $t_{(j-1)}$. Then the estimate of $\lambda_0(t)$ for the interval $(t_{(j-1)}, t_{(j)}]$, is:

$$\hat{\lambda}_j = \frac{d_j}{(t_{(j)} - t_{(j-1)}) \cdot \sum_{i \in R_j} e^{\beta' \cdot x_i}}$$

where d_j is the number of failures which occurred in the j^{th} time interval. Breslow's estimator of the cumulative baseline hazard at time t is:

$$\hat{\Lambda}_0(t) = \sum_{t_{(j)} \leq t} \frac{d_j}{\sum_{i \in R_j} e^{\beta' \cdot x_i}}$$

and the baseline hazard itself is:

$$\hat{\lambda}_0(t_{(j)}) = \frac{d_j}{\sum_{i \in R_j} e^{\beta' \cdot x_i}} \quad (3.5)$$

calculated at each failure time $t_{(j)}$.

3.2 Schoenfeld residuals [Schoenfeld, 1982]

For a Cox-PH model involving K covariates for each subject i : $x_i = (x_{1i}, x_{2i}, \dots, x_{Ki})$ and $\beta = (\beta_1, \beta_2, \dots, \beta_K)$, Schoenfeld residuals are defined at each failure time $t_{(j)}$ as the difference between the covariates of the subject who fails at time $t_{(j)}$ and their estimated values, given the subjects still at risk at time $t_{(j)}$:

$$\hat{r}_j = x_j - \hat{E}(x_j | R_j) \quad (3.6)$$

There is one Schoenfeld residual for each failure time (assuming that only one individual fails at a time) and each residual is a vector of K components (one for each covariate), where $\hat{E}(x_j | R_j)$ is obtained by substituting the maximum likelihood estimates $\hat{\beta}$ in $E(x_j | R_j)$.

$$E(x_j | R_j) = \frac{\sum_{k \in R_j} x_j \cdot e^{\beta' x_k}}{\sum_{k \in R_j} e^{\beta' x_k}}$$

Note that those residuals belong to time points and cannot be computed for all individuals. It can be shown, that under the correct model the values of \hat{r}_j are asymptotically uncorrelated with their expected value zero.

Each component \hat{r}_{kj} corresponds to the k^{th} covariate. A plot of the \hat{r}_{kj} values against time would show possible departures from the PH assumption related to the k^{th} covariate (Figure 3.1). To discover the presence of patterns over time (hence departure from the PH assumption) one would need to smooth the residuals. Figure 3.1 shows no change of the residual pattern over time, hence we can conclude that covariate LVEF satisfies the PH assumption.

If more than one subject fails at time $t_{(j)}$, a separate residual is calculated for each subject instead of averaging (Therneau & Grambsch, 2000).

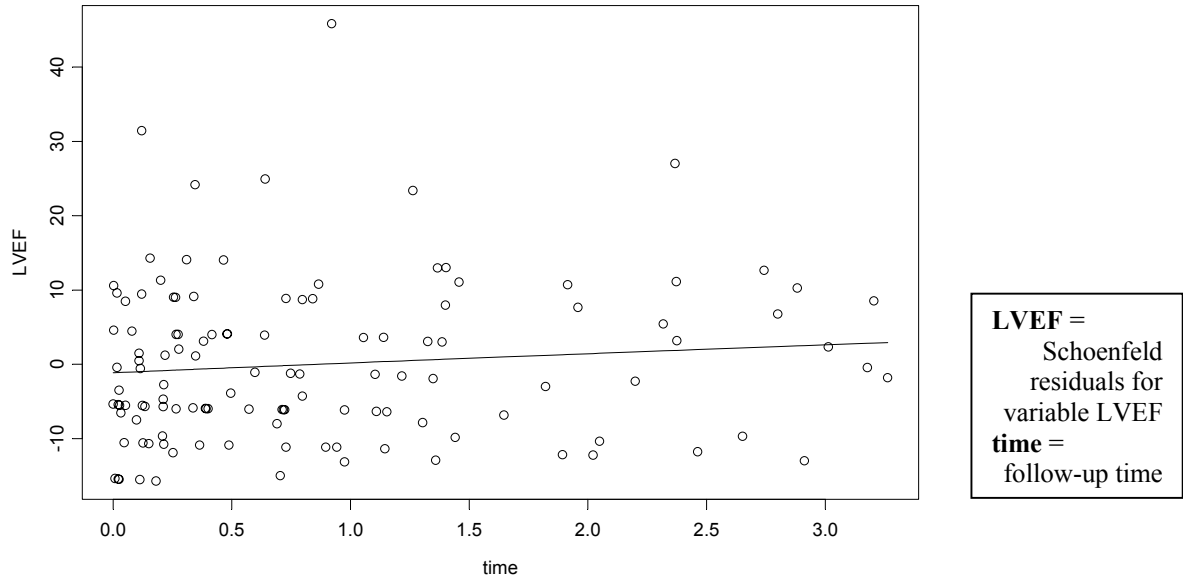


Figure 3.1: *Scatter plot of the Schoenfeld residuals for factor LVEF against follow-up time for the prognostic model (section 10.4.1) built on the EMIAT data. The residuals are smoothed with cubic splines.*

3.3 Martingale residuals [Barlow & Prentice, 1988]

The basis for development of martingale-type residuals is the difference of the counting process observed on individual i and the integrated intensity function for this counting process:

$$M_i(t) = N_i(t) - \int_0^t Y_i(s) \cdot e^{\beta' \cdot Z_i(s)} d\Lambda_0(s) \quad (3.7)$$

where:

$i = 1, \dots, n$

$Y_i(t)$ is a 0-1 process, indicating whether the i^{th} subject is at risk at time t

β is a vector of regression coefficients

$Z_i(t)$ is a p-dimensional vector of covariate processes

Λ_0 is the baseline cumulative hazard function.

$N_i(t)$ is the counting process for subject i ; for right censoring, it is 0 prior to time of event and 1 thereafter.

We are in the framework of the Cox-PH model where Λ_0 is unspecified and $Y_i(t) = 1$ until the first event or censoring and zero thereafter. For example, if patient i is censored or fails at time $t = 5$,

$$Y_i(t) = \begin{cases} 1 & \text{if } t \leq 5 \\ 0 & \text{if } t > 5 \end{cases}$$

Subject to standard measurability and integrability requirements, $M_i(\cdot)$ will be a subject-specific martingale [Gill, 1980, 1984].

Using standard partial likelihood theory, we can get the maximum likelihood estimate of β and consequently an estimate of Λ_0 [Breslow, 1974]:

$$\hat{\Lambda}_0(t) = \int_0^t \frac{\sum dN_i(s)}{\sum Y_i(s) \cdot e^{\hat{\beta}' \cdot Z_j(s)}} \quad (3.8)$$

We can define the martingale residuals as:

$$\hat{M}_i(t) = N_i(t) - \int_0^t Y_i(s) \cdot e^{\hat{\beta}' \cdot Z_i(s)} d\hat{\Lambda}_0(s) \quad (3.9).$$

Martingale residuals have the following properties:

1. $E(\hat{M}_i) = \text{cov}(\hat{M}_i, \hat{M}_j) = 0$, asymptotically
2. $\sum \hat{M}_i(t) = 0, \quad \forall t$

Specifically for Cox-PH model, the definition of martingale residuals reduces (see Appendix A) to:

$$\hat{M}_i = \delta_i - \hat{\Lambda}_0(t_i) \cdot e^{\hat{\beta}' \cdot Z_i} = \delta_i - \hat{\Lambda}_i(t_i, Z_i) \quad (3.10)$$

where:

t_i is the observation time for subject i and

δ_i is the final status for subject i .

The residuals can be interpreted at each time t as:

"... the difference over $[0, t]$ of the observed number of events minus the expected number given the model, or as excess deaths." (Therneau et al., 1990)

Kay (1977) came to this residual from a different perspective and Crowley & Hu, 1977 developed a similar residual based on the original work of Cox & Snell, 1968.

Notice that since the status can take only values of 0 or 1 and the hazard is always non-negative, the martingale residual for the Cox-PH model takes values only in the interval $(-\infty, 1]$. The following figures 3.2 & 3.3 give an example of what a plot of the martingale residuals could look like and a box plot for demonstration of the skewed nature of their distribution.

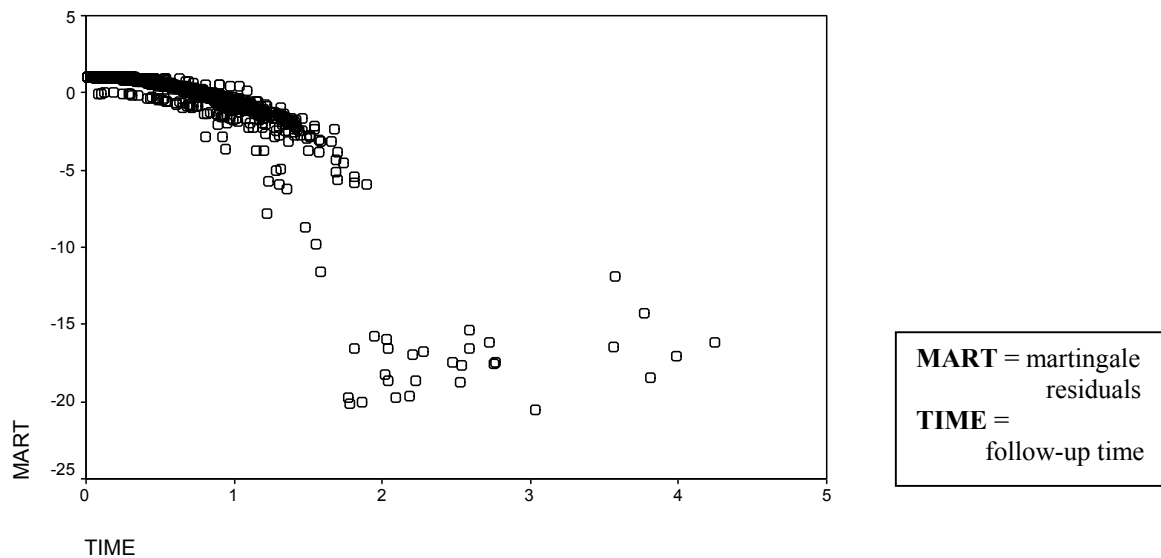


Figure 3.2: *Scatter plot of the martingale residuals against follow-up time for a model on a data set simulated as in chapter 9.*

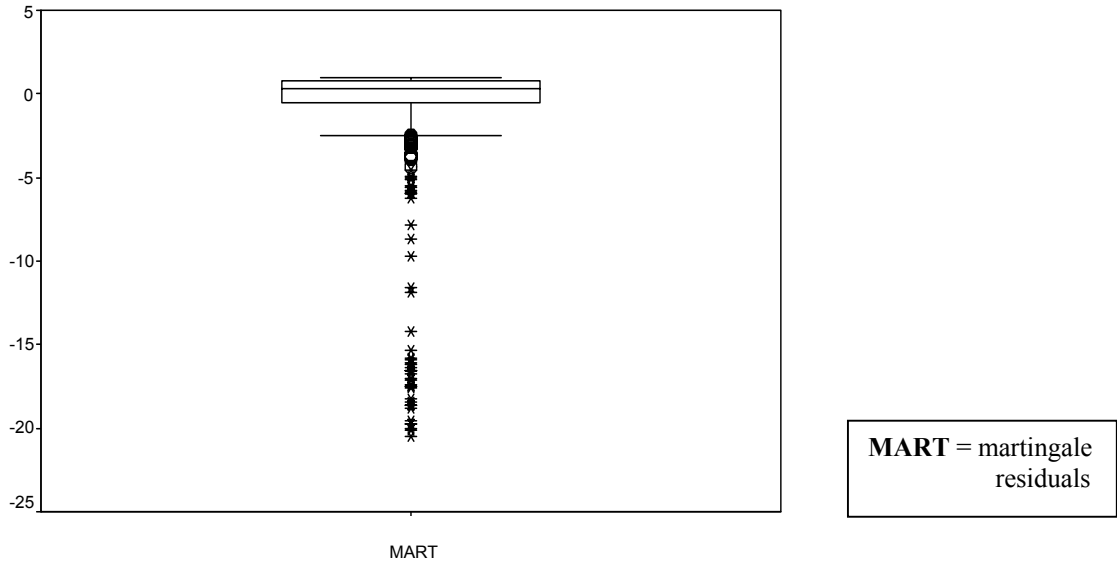


Figure 3.3: *Box-and-whisker plot for the martingale residuals from figure 3.2.*

3.4 Score residuals [Therneau et al, 1990]

The score residuals are a martingale-transform type of residuals defined for each subject on the basis of his/her contribution to the score statistic (3.11). In our case the integral is with respect to the martingale residual, which involves β and A_0 . When A_0 is unspecified, as it is the case in the Cox-PH model, it can be estimated using Breslow's estimate. Then, when $\beta = b$, the derivative of the partial likelihood function L_p with respect to β_j can be written as:

$$\begin{aligned}
 \left[\frac{\partial \ln L_p}{\partial \beta_j} \right]_{\beta=b} &= \sum_{i=1}^n \int_0^{\infty} [Z_{ij}(s) - \bar{Z}_j(b, s)] dN_i(s) \\
 &= \sum_{i=1}^n \int_0^{\infty} [Z_{ij}(s) - \bar{Z}_j(b, s)] d\hat{M}_i(s) \\
 &= \sum_{i=1}^n L_{ij}(b, \infty)
 \end{aligned} \tag{3.11}$$

where $\bar{Z}_j(b, s)$ is the weighted mean of the covariates over the risk at time s :

$$\bar{Z}_j(b, s) = \frac{\sum_{i=1}^n Y_i(s) \cdot e^{b' \cdot Z_i(s)} \cdot Z_{ij}(s)}{\sum_{i=1}^n Y_i(s) \cdot e^{b' \cdot Z_i(s)}} \quad (3.12)$$

We define $L_{ij}(\hat{\beta}, \cdot)$ as the score process and $L_{ij}(\hat{\beta}, \infty)$ as the score residual of the i^{th} subject and the j^{th} variable. Score residuals measure the leverage exerted by each subject on parameter estimates in that they provide an estimate of the changes in the coefficients β that would occur when each of the observations are deleted. Therefore, one should look for outliers in the residual plots. Several outliers are visible on the score residual plots of figure 3.4. Each outlier corresponds to a patient, who's value for the factor being analyzed influence strongly the model coefficient of that factor.

The score residuals also sum to zero.

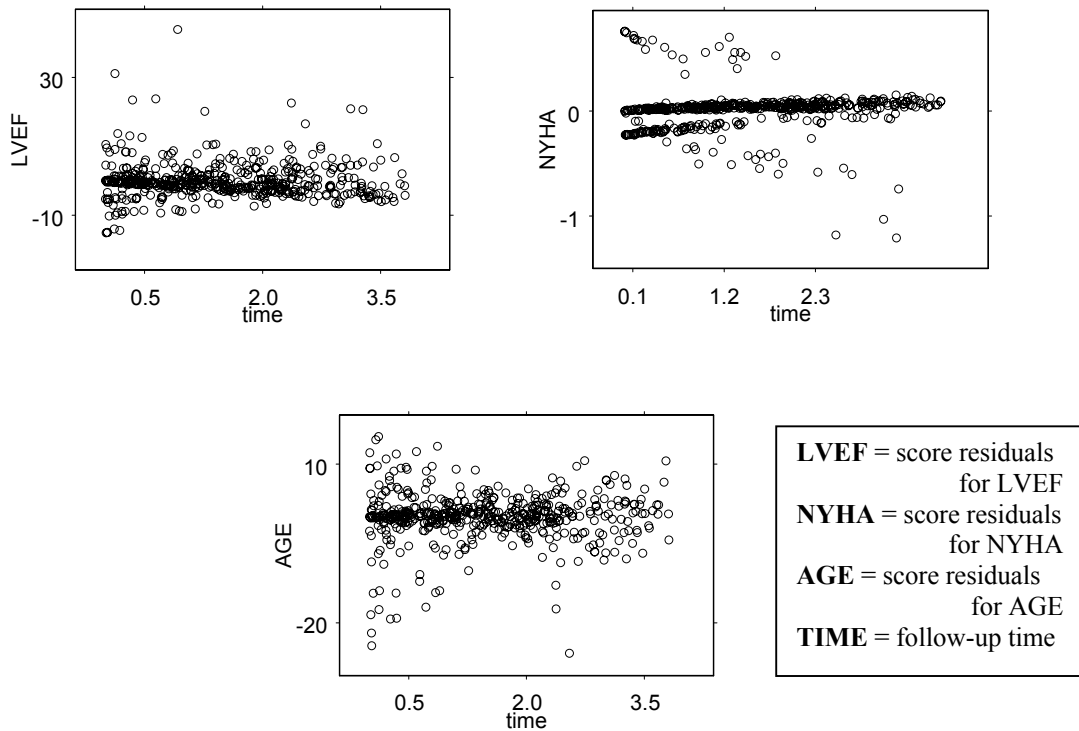


Figure 3.4: *Scatter plots of the score residuals of factors LVEF, NYHA, and AGE against follow-up time for the model from section 10.4.1.*

3.5 Deviance residuals (Therneau et al, 1990)

A major deficiency of the martingale residuals is their skewed distribution. They take values in the interval $(-\infty, 1]$. A transformation helps solving this problem:

Define: deviance:

$$D = 2 \cdot [LL(saturated) - LL(\hat{\beta})] \quad (3.13),$$

where the saturated model is one for which β is free, i.e. each observation i has its own vector of coefficients β . Any nuisance parameters, such as the baseline hazard Λ_0 are held constant across the two models. Assuming known Λ_0 and letting h_i be the individual estimates of β for each subject i :

$$D = 2 \cdot \sup_h \sum \left\{ \int \left(\ln e^{h_i' \cdot Z_i} - \ln e^{\hat{\beta}' \cdot Z_i} \right) dN_i(s) - \int Y_i(s) \cdot \left(e^{h_i' \cdot Z_i} - e^{\hat{\beta}' \cdot Z_i} \right) d\Lambda_0(s) \right\} \quad (3.14)$$

Using Lagrange multiplier, the maximal value of h_i satisfies:

$$\int_0^\infty Y_i(s) \cdot e^{h_i' \cdot Z_i} d\Lambda_0(s) = \int_0^\infty dN_i(s)$$

Let the Martingale residual with estimated β and known Λ_0 be:

$$\tilde{M}_i(t) \equiv N_i(t) - \int_0^t e^{\hat{\beta}' \cdot Z_i} d\Lambda_0(s) \quad (3.15)$$

Substituting in the deviance definition:

$$\begin{aligned} D &= -2 \sum \left[\tilde{M}_i + \ln \left(\frac{e^{\hat{\beta}' \cdot Z_i}}{e^{h_i' \cdot Z_i}} \right) \cdot \int dN_i(s) \right] \\ &= -2 \sum \left[\tilde{M}_i + N_i(\infty) \cdot \ln \left(\frac{N_i(\infty) - \tilde{M}_i}{N_i(\infty)} \right) \right] \end{aligned}$$

Estimation of Λ_0 results in the replacement of \tilde{M}_i by \hat{M}_i in the formula. The deviance residual is the signed square root of the deviance D. It is zero if and only if $\hat{M}_i = 0$. For the Cox-PH model, the deviance simplifies to:

$$d_i = \text{sgn}(\hat{M}_i) \cdot \sqrt{-2 \cdot [\hat{M}_i + \delta_i \cdot \ln(\delta_i - \hat{M}_i)]} \quad (3.16)$$

The $\ln(\cdot)$ function inflates martingale residuals close to 1 and the square root contracts the large negative values.

Usually deviance residuals are plotted against the risk score of the model ($\beta \cdot x$), but for the purpose of outlier screening one can use observation time instead of score or simply a box-and-whiskers plot. The martingale residuals used in figures 3.2 & 3.3 were transformed to deviance residuals in figures 3.5 & 3.6. Notice that the magnitude of the residuals changes, but their relationship to one another stays the same since they are monotonic transformations of each other (see figure 3.7 for illustration). The same negative outliers can be detected visually on the scatter plots. As expected, after the transformation the distribution of the deviance residuals appears more symmetric and differences among the positive residuals are enhanced (recall: martingale residuals can be at most 1 whereas deviance residuals do not have such restrictions).

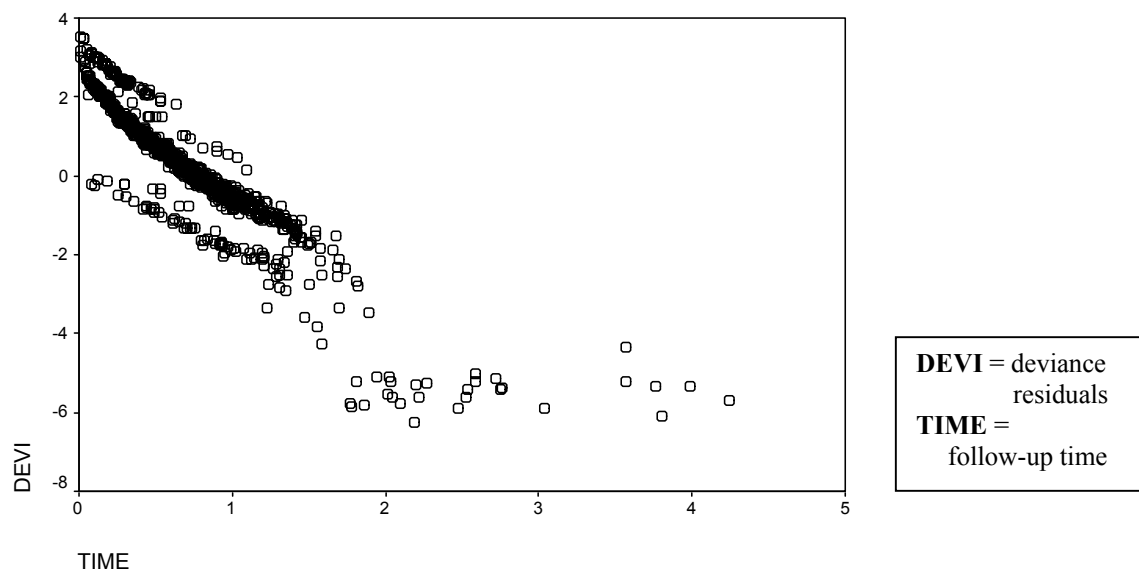


Figure 3.5: *Scatter plot of the deviance residuals against follow-up time corresponding to the martingale residuals from figure 3.2.*

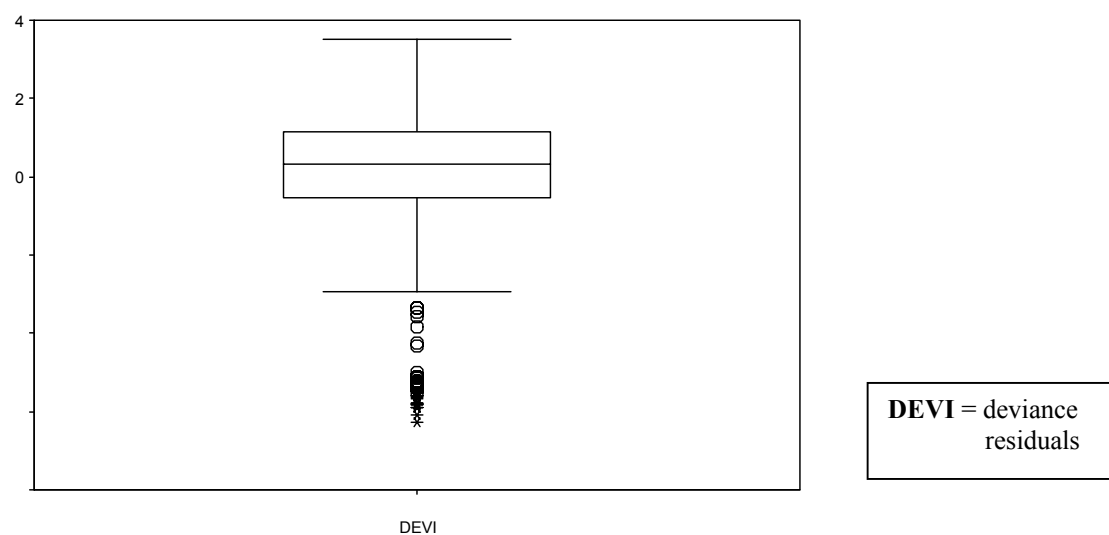


Figure 3.6: *Box-and-whiskers plot of the deviance residuals from figure 3.5.*

3.6 Log-odds and normal deviate residuals (Nardi & Schemper, 1999)

Nardi & Schemper (1999) suggested the use of log-odds or normal deviate residuals for outlier screening, which seem to have better distributional properties than the Deviance residuals. The log-odds and normal deviate residuals are constructed on the basis of the estimated survival function for individual i and his/her observed event time t_i or censored time t_i^c . The prediction of survival is considered "perfect" if $\hat{S}_i(t_i) = 0.5$. Those residuals do not measure directly the difference between observed and predicted survival time. Instead, they do this indirectly by comparing the estimated survival probability at time t_i (or t_i^c) with the "perfect" value of 0.5.

Log-odds residuals are defined as the logit transformation of $\hat{S}_i(t_i)$:

$$l_i = \begin{cases} \log \frac{\hat{S}_i(t_i)}{1 - \hat{S}_i(t_i)} \\ \log \frac{\hat{S}_i(t_i^c)}{2 - \hat{S}_i(t_i^c)} \end{cases} \quad (3.17)$$

Normal deviate residuals are defined as the probit transformation of $\hat{S}_i(t_i)$:

$$n_i = \begin{cases} \Phi^{-1}\left\{\hat{S}_i(t_i)\right\} \\ \Phi^{-1}\left\{\frac{\hat{S}_i(t_i^c)}{2}\right\}, \end{cases} \quad (3.18)$$

where Φ is the normal cumulative distribution function.

The censored case in both definitions deserves special attention. Since for censored individuals the censoring time is always less than the time of event, a uniform

distribution of $\hat{S}_i(T_i)$ is assumed on the interval $[0, \hat{S}_i(t_i^c)]$ (Crowley & Hu, 1977). The unknown value of $\hat{S}_i(t_i)$ is then replaced by its mean $\frac{\hat{S}_i(t_i^c)}{2}$.

3.7 Suitable residuals for responder identification

For the purpose of responder identification we need residuals to the Cox-PH model which correspond to data points and are not explicitly connected to single prognostic factors contained in the model. Such residuals would be able to identify outlying points with poorly predicted individual outcomes by the prognostic model. Those points can be used for predictive factor identification and, ultimately, responder identification purposes.

Schoenfeld residuals check the validity of the proportional hazards assumption. A large (positive or negative) residual indicates that the event which occurred at time t_j is unlikely under the current model, given the covariates of the individual who failed relative to those still at risk. They belong to time points rather than to individuals and are plotted against the prognostic factors. Therefore, they are not suitable for responder identification.

Score residuals look for presence of observations which are influential to a prognostic factor's coefficient estimate. They are different for each prognostic factor and, thus, cannot be used for predictive factor identification.

Martingale residuals, on the other hand, are suitable for responder identification. Naturally, they were originally constructed with a different implementation in mind, namely, to deal with the linearity assumption for prognostic factors, as summarized in table 3.1 (Harrell, 2001), but their properties together with their simple and logical interpretation make them a natural candidate for responder (outlier) identification.

Table 3.1: *Use of martingale residuals (Harrell, 2001).*

Purpose	Method
Estimate transformation for a single variable	Force $\hat{\beta}_1 = 0$ and compute residuals from the null regression
Check linearity assumption for a single variable	Compute $\hat{\beta}_1$ and the residuals from the linear regression (ordinary Cox model)
Estimate martingale transformations for p variables	Force $\hat{\beta}_1, \dots, \hat{\beta}_p = 0$ and compute residuals from the global null model
Estimate transformations for variable i adjusted for the rest $p-1$ variables	Estimate $p-1$ β 's, forcing $\hat{\beta}_i = 0$. Compute residuals from mixed global/null model

Martingale residuals for the Cox-PH model are defined in 3.10 to be the difference of the censoring indicator and the estimated hazard rate at each observation. Recall that martingale residuals can only achieve values in the interval $(-\infty, 1]$, since δ_i switches only between the values of 0 and 1. For residual interpretation purposes the censoring indicator can be thought of as a classification rule, which places patients into either the low or the high hazard group. This results in only a few possible scenarios:

Martingale residuals with values close to zero:

As by most other residuals, values around zero reflect good fit of the model. In our situation this can be achieved if $\delta_i = 1$ and $\hat{\Lambda}_i \approx 1$, which means that the i^{th} patient with an event was predicted to be at high risk, or if $\delta_i = 0$ and $\hat{\Lambda}_i \approx 0$, which means that the i^{th} patient was censored and predicted to be at low risk. Those are candidates for **non-responders**.

$$\left. \begin{array}{l} \delta_j = 0 \\ \hat{\Lambda}_j \approx 0 \end{array} \right\} \Rightarrow \hat{M}_j \approx 0 \qquad \left. \begin{array}{l} \delta_j = 1 \\ \hat{\Lambda}_j \approx 1 \end{array} \right\} \Rightarrow \hat{M}_j \approx 0$$

Martingale residuals with large positive values:

Large values of any residuals are a sign of a bad fit of the prognostic model and here – a possible sign of existing predictive factors. Values of the martingale residuals close to 1 can be achieved only if $\delta_i = 1$ and $\hat{\Lambda}_i \approx 0$, i.e. the i^{th} patient was predicted to be at low risk but he/she died. Such patients are candidates for **negative responders**.

$$\left. \begin{array}{l} \delta_j = 1 \\ \hat{\Lambda}_j \approx 0 \end{array} \right\} \Rightarrow \hat{M}_j \approx 1$$

Martingale residuals with large negative values:

Large negative values are also a sign of a bad fit. Large negative values of the martingale residuals (e.g. -1) are achieved if $\delta_i = 0$ and $\hat{\Lambda}_i > 0$ (e.g. $\hat{\Lambda}_i \approx 1$), i.e. the i^{th} patient was predicted to be at high risk but he/she was censored (i.e. did better than expected from the prognostic model). Such patients are candidates for **positive responders**. A large negative martingale residual is also possible for patients who died and have extremely large predicted hazard rate (e.g. $\delta_i = 1$ and $\hat{\Lambda}_i \approx 2$). Notice, that even though the patient dies, he would still be candidate for a positive responder, since in order to have such a large hazard rate, he must have lived much longer than expected.

$$\left. \begin{array}{l} \delta_j = 0 \\ \hat{\Lambda}_j > 0 (\approx 1) \end{array} \right\} \Rightarrow \hat{M}_j < 0 (\approx -1) \qquad \left. \begin{array}{l} \delta_j = 1 \\ \hat{\Lambda}_j > 1 (\approx 2) \end{array} \right\} \Rightarrow \hat{M}_j < 0 (\approx -1)$$

As mentioned in chapter 3.2, martingale residuals have two major disadvantages. They take values only in the interval $(-\infty, 1]$ and their distribution is highly asymmetric. Furthermore, censored cases always have negative residuals, which skews the distribution even further for data sets with large percent censoring. Deviance residuals were created especially to deal with the first problem and they have better distributional properties (Therneau et al, 1990). However, they do not always manage to transform the martingale residuals to a symmetric distribution, especially for data with large percentage of censoring. Although the definition of deviance residuals (3.16) looks

rather complicated, they can just as well as the martingale residuals be interpreted in the traditional form of *expected – predicted value* (for details, see Appendix A). If we regard once again the censoring indicator as the expected value from the data and the estimated hazard rate – as the predicted value from the (prognostic) model, with some calculations we reach the following conclusions:

Deviance residuals with values close to zero:

If the expected and the predicted values are the same, the resulting residual is small in absolute value or zero. Just as by martingale residuals, this can be achieved if a patient is predicted (prognostic model) to have low hazard and he/she is censored or if a patient is predicted to have high hazard (about equal to one) and he/she experiences an event. Such patients are candidates for **non-responders**.

$$\left. \begin{array}{l} \delta_j = 0 \\ \hat{\Lambda}_j \approx 0 \end{array} \right\} \Rightarrow d_j \approx 0 \qquad \left. \begin{array}{l} \delta_j = 1 \\ \hat{\Lambda}_j \approx 1 \end{array} \right\} \Rightarrow d_j \approx 0$$

Deviance residuals with large positive values:

Deviance residuals are transformed martingale residuals and as such, they can have positive values only if an event occurred and the predicted hazard rate is less than one. Unlike martingale residuals, they do not have an upper limit of one (see figure 3.5). Patients with large positive values of the residuals are candidates for **negative responders**:

$$\left. \begin{array}{l} \delta_j = 1 \\ \hat{\Lambda}_j < 1 (\approx 0.5) \end{array} \right\} \Rightarrow d_j > 0 (\approx 2)$$

Deviance residuals with large negative values:

Deviance residuals can have negative values both for censored cases and for cases with events. If a patient is predicted to have large hazard but he/she is censored, the residual would be large negative. The patient shows improvement under the new treatment (he/she is censored before an event occurred) in comparison to his/her predicted hazard

(large positive) as expected from the prognostic model developed on the classical treatment group. Just as by martingale residuals, the patient can have an event and still show improvement under the new treatment, if his/her predicted hazard is much larger than one. Such situation would show that the patient did experience an event, but much later than expected. Such patients are candidates for **positive responders**.

$$\left. \begin{array}{l} \delta_j = 0 \\ \hat{\Lambda}_j > 0 (\approx 2) \end{array} \right\} \Rightarrow d_j < 0 (\approx -2) \qquad \left. \begin{array}{l} \delta_j = 1 \\ \hat{\Lambda}_j > 1 (\approx 4.5) \end{array} \right\} \Rightarrow d_j < 0 (\approx -2)$$

The general relationship between the size of the predicted hazard and the resulting residual (martingale and deviance) is plotted in figure 3.7 for censored and uncensored cases. And the relationship between martingale and deviance residuals (defined in 3.16) is plotted in figure 3.8. As expected, those residuals are highly correlated (Spearman's rho correlation coefficient = .998, $p < .001$).

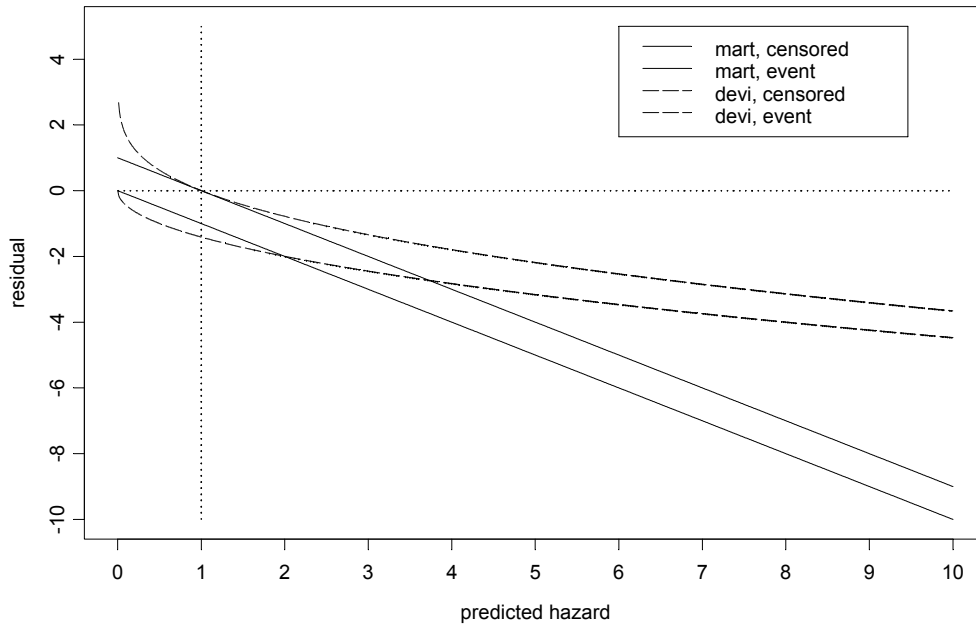


Figure 3.7: *Relationship between the predicted hazard and the martingale and deviance residuals for the event and censored (always non-positive) cases. The relationship between predicted hazard and martingale residual is linear, whereas deviance residuals transform that relationship.*

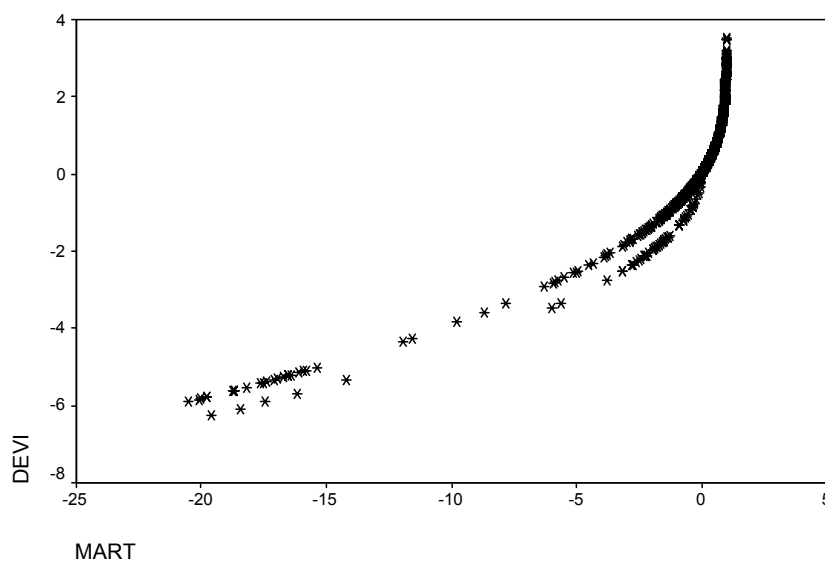


Figure 3.8: *Relationship between the deviance and martingale residuals from figures 3.5 and 3.2.*

The log-odds or normal deviate residuals, which can be used for outlier screening, have better distributional properties than the deviance and the martingale residuals. However, they are both highly correlated (Spearman's rho correlation coefficient = .998, $p < .001$) with the martingale residuals (see figures 3.9 & 3.10).

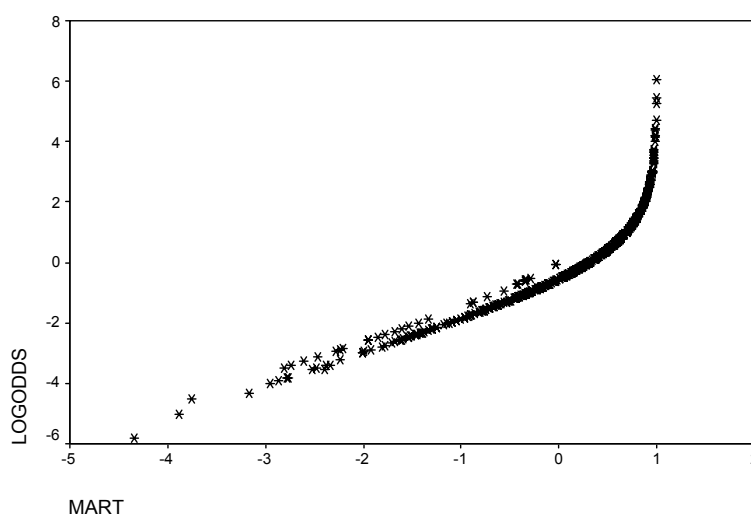


Figure 3.9: *Relationship between the martingale (MART) and log-odds (LOGODDS) residuals on a model from the simulation study in chapter 9.*

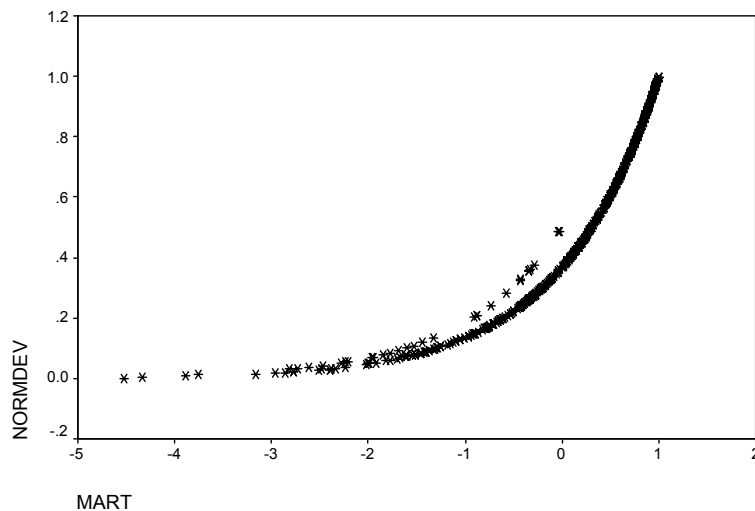


Figure 3.10: *Relationship between the martingale (MART) and normal-deviate (NORMDEV) residuals corresponding to figure 3.9. Notice, that the log-odds and normal-deviate residuals from figures 3.9 & 3.10 were calculated on a different data set than the residuals in figure 3.8, hence the different scale of the martingale residuals.*

In addition, even though the log-odds and normal deviate residuals quantify the departure from perfect prediction of the prognostic model, they cannot be interpreted in the classical expected vs. predicted form, which makes them undesirable for responder identification.

The above residual overview leaves us with the following thought. We can either have interpretable, hence usable for responder identification residuals, which have distributional problems connected to censoring, or we can choose the improved residuals, which are perfect for outlier screening and with nice symmetric distributions, but which are unusable as responder identifiers since they cannot be interpreted in the *expected – predicted* form (Nardi & Schemper, 1999). This leaves us with the martingale and the deviance residuals as suitable for responder identification purposes. Since they are highly correlated with each other and since they basically identify the same groups of outliers (see figure 3.2 & 3.5), a simulation study is needed in order to choose the more appropriate residual for responder identification (see chapter 9).

4. THE CLASSICAL APPROACH: COX-PH MODEL WITH INTERACTIONS

Up to now, the classical approach for responder identification in clinical trials has been the Cox-PH model including interaction terms between the treatment and some or all of the covariates (Schemper, 1998).

4.1 Definition

In a clinical trial with two arms, in which a classical and a new treatment are compared, one would use the following version of the Cox-PH model on the entire data set:

$$\lambda(t, x_i, z_i) = \lambda_0(t) \cdot e^{\overbrace{\beta'_x \cdot x_i}^{\text{prognostic}} + \overbrace{\beta'_I \cdot z_i \cdot \text{treat} + \beta'_z \cdot z_i + \beta_T \cdot \text{treat}}^{\text{predictive}}}$$

where:

- i – patient identifier, $i = 1, \dots, n$
- x – vector of prognostic factors
- β_x – vector of coefficients of the prognostic factors
- z – vector of predictive factors ($z \subset x$ is possible)
- β_z – vector of coefficients of the predictive factors;
to avoid double appearance, $\beta_z[i] = 0$ for $z_i \subset x$
- treat – factor indicating treatment group (0 = classical, 1 = new)
- β_T – coefficient of treatment indicator
- β_I – vector of coefficients of the interaction terms

4.2 Responder identification

If a certain predictive factor interaction term shows to be adding information to the model, this should be interpreted as follows:

- If the coefficients in the predictive part of the model are such, that the presence of factor z_i in the model increases the hazard of patients having that factor and taking the new treatment, we can say that z_i is a predictive factor and patients having this characteristic are **negative responders** of the new treatment (see figure 2.2).
- Naturally, if the coefficients in the predictive part of the model lead to reduction of the hazard in the presence of factor z_i , then z_i would be a predictive factor which defines the **positive responder** group.

The problem with this method is, that in order for it to recognize a combination of factors as predictive, this particular combination has to be present in the model as interaction. Even assuming that the interaction between the factors is linear, the order of the interaction term is unknown. If two predictive factors and factor treatment should show interaction, one needs to consider all possible interaction terms of up to third order in order to give a chance of a covariate selection procedure to choose the right combination. The number of possible interaction terms to be considered grows rapidly as the number of factors grows. It is also known, that the power of stepwise variable selection procedures decreases as the number of variables (variable combinations) increases.

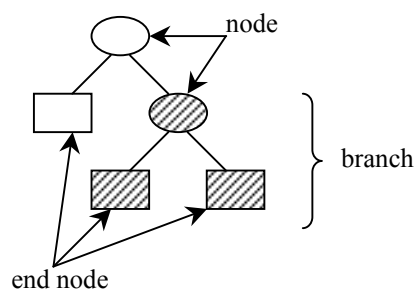
Considering the limitations of this simple approach, it is clear, that a new more involved exhaustive method is needed. A class of such methods is suggested and discussed in chapters V through VIII.

5. CLASSIFICATION AND REGRESSION TREES (CART) – RECURSIVE PARTITIONING

We will use the following standard tree terminology:

Tree	A model based on recursive (usually binary) partitioning
Node	A position in the tree where a new partitioning can be performed. The node is the current space for all immediately following operations.
Split	Position in the tree where the current space (node) is partitioned into (two) subspaces
Root tree	A tree consisting of one node and no splits, i.e. the original space.
End node (leaf)	A node on which no more splits can be performed. The objects in each leaf are estimated with an appropriate function.
Branch	A node with all its following splits and nodes. A branch is a subtree.
Pruning	Cutting off branches.
Classification tree	A tree model appropriate for data with a categorical response variable or, after alteration, for survival response.
Regression tree	A tree model appropriate for data with a continuous response variable.

Tree diagram



Although the original clinical trial data in our problem is survival data (two response variables: survival time and censoring indicator), predictive factor identification does not require application of survival trees. A survival model (in our case Cox-PH model) was already used on the data for prognostic factor identification. From this point on, we will have a single response variable – the residuals to the prognostic model.

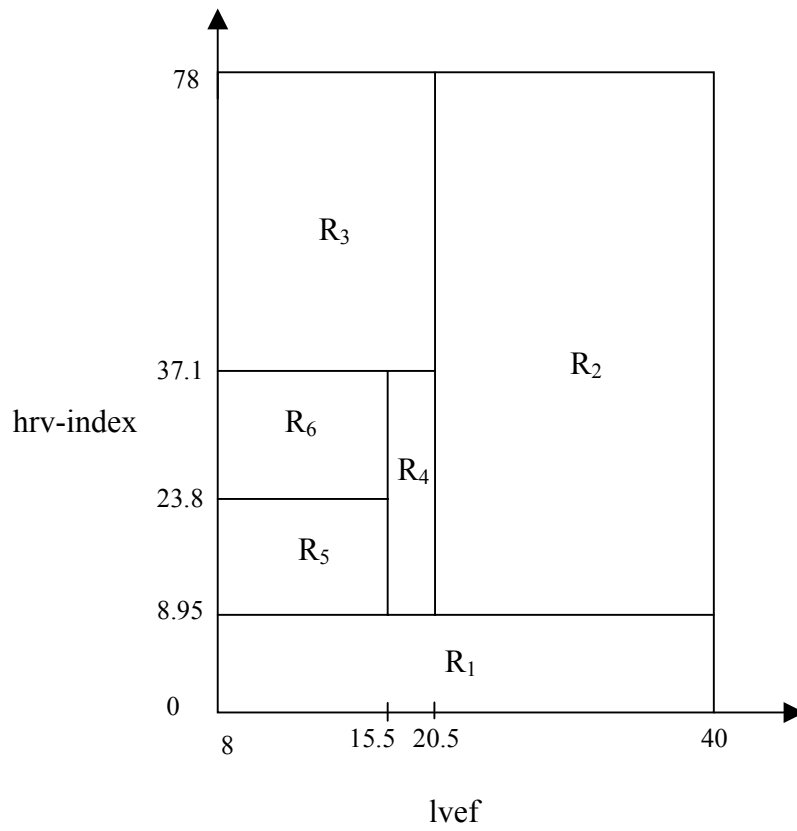
The regression and classification tree method (Breiman et al, 1984) is a method which employs recursive partitioning in order to split the response space into a set of rectangles. In classification trees, the response is categorical and the objects in one rectangle would be predicted to be in one of the response categories (classes). Since martingale and deviate residuals are continuous, we will need regression trees, which fit a simple (e.g. constant) model in each of the resulting rectangles to predict the response of the points in them.

For illustration purposes, it is convenient to use just two continuous factors and a continuous response. Factors LVEF and HRVI were chosen from the treatment arm of the EMIAT data set for this example. The response variable here is the martingale residual of the prognostic model (see chapter 10.4.1 for details on the prognostic model). A binary regression tree was built on the data (figure 5.1b), which splits the input space recursively in two parts, as shown in figure 5.1a. The first split is made at the value of 8.95415 of factor HRVI, the second – at value 20.5 of LVEF, and so on. The result of this recursive binary partitioning is a set of 6 regions R_1, \dots, R_6 . The regression model predicts the residuals in each region with a constant c_m (Hastie, Tibshirani, Friedman, 2001):

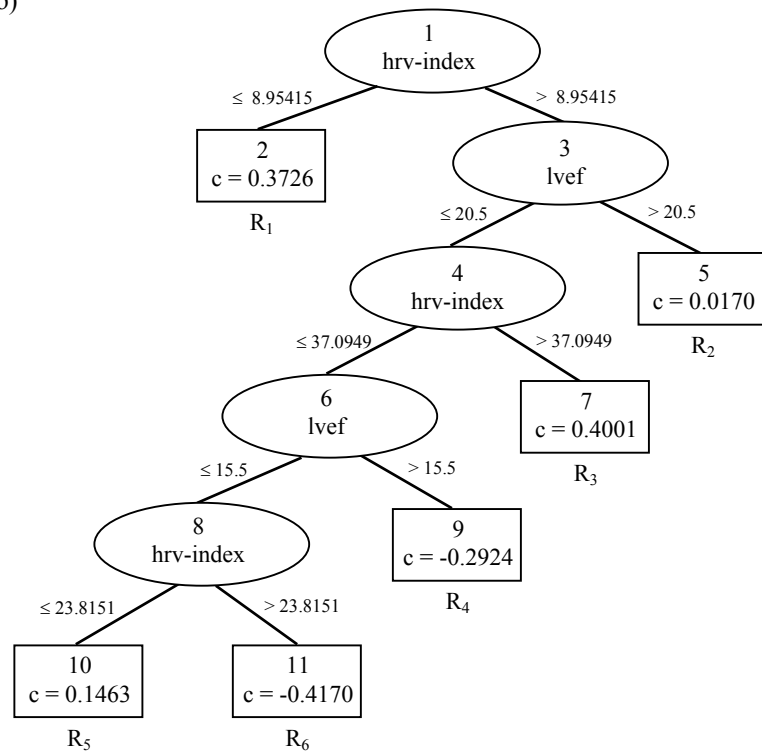
$$\hat{f}(x) = \sum_{m=1}^6 c_m \cdot I\{(x_1, x_2) \in R_m\},$$

where x_1 & x_2 are the two input factors. The resulting regions are represented in the 3D plot of figure 5.1c and the tree model representing this series of splits is shown in figure 5.1b, where the end nodes of the tree represent the 6 regions.

a)



b)



c)

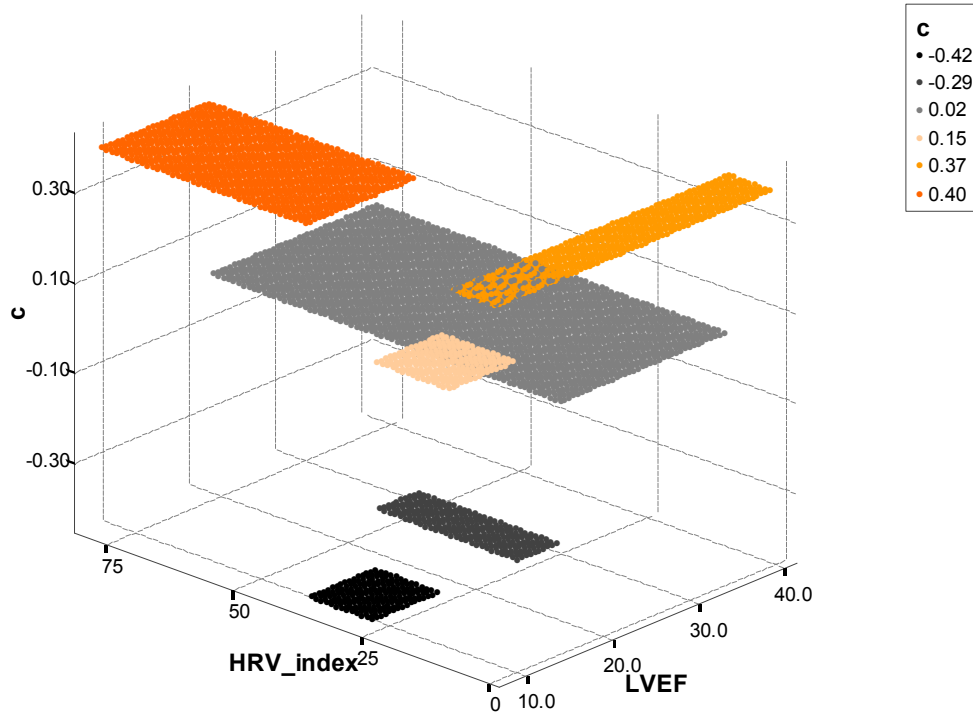


Figure 5.1: *A regression tree example built on a two dimensional input space from the EMIAT data set, including factors LVEF & HRV_index (HRVI), with the martingale residuals of the Cox-PH model from section 10.4.1 as a response variable.*

- a) *The two dimensional input space, divided into six regions as defined by the end nodes of the regression tree in b).*
- b) *The regression tree model*
- c) *Three dimensional representation of a) including the mean response (c) for each region, which represents the mean of the martingale residuals for patients in the six regions of a).*

Notice that the predictors x_i may also be categorical, in which case the set of possible splits is predefined and finite (for binary predictors there is only one possible split point). Also, in general, we have more than two predictors to choose from.

5.1 Growing a regression tree

Consider the following data available for growing a tree, the so called learning sample L : a set of p predictors $x_i = (x_{i1}, x_{i2}, \dots, x_{ip})$ and a response y_i , available as pairs (x_i, y_i) for a total of N observations ($i = 1, \dots, N$). The tree algorithm must be able to select splitting variables and corresponding split points, starting with L and ending when a stopping criteria is reached. The algorithm must also decide at each node if this is an end node or if further splitting is needed. Additionally, one may choose to reconsider the tree architecture and collapse some unimportant nodes, using pruning.

5.1.1 Splitting

Splitting in regression trees, as introduced by Breiman et al (1984), is done with the help of least squares regression. If a tree partitions the space into M regions R_1, \dots, R_M , which are modeled with the simplest possible regression model – a constant c_m , then the response variable can be described by:

$$y = f(x) = \sum_{m=1}^M c_m \cdot I(x \in R_m)$$

Regression tree models, as described by Breiman et al (1984), use the least mean squared error of $f(x)$ in the region as a splitting criterion. Each possible split defined with splitting variable x_i and split point t_i is evaluated by calculating the mean squared error of predictor $f(x_i)$ of y_i :

$$\frac{1}{N} \sum_{i=1}^N (y_i - f(x_i))^2$$

That means that the best constant \hat{c}_m for region R_m is just the average of y_i in R_m :

$$\hat{c}_m = \text{ave}(y_i \mid x_i \in R_m)$$

The optimal cutpoint is chosen by minimizing the expected mean squared error. In terms of splitting L in two regions at predictor j and point t_s , if R_1 and R_2 are the two resulting subspaces at this split, then:

$$R_1(X_j, t_s) = \{X \mid X_j \leq t_s\} \quad \text{and} \quad R_2(X_j, t_s) = \{X \mid X_j > t_s\}$$

Then one should be looking for splitting variable x_j and point t_s , which minimize:

$$\min_{x_j, t_s} \left[\min_{c_1} \sum_{x_i \in R_1(x_j, t_s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(x_j, t_s)} (y_i - c_2)^2 \right]$$

The minimization over c_1 and c_2 is solved by the Bayes predictors (see Breiman et al, 1984, for extensive theory of the splitting process):

$$\hat{c}_1 = \text{ave}(y_i \mid x_i \in R_1(x_j, t_s)) \quad \text{and} \quad \hat{c}_2 = \text{ave}(y_i \mid x_i \in R_2(x_j, t_s))$$

Thus, minimization over x_j and t_s is done by considering all possible x_j and t_s combinations in L , and taking the one which minimizes the sum of the averages of y_i in the two resulting subspaces.

Splitting is done iteratively, starting with the original data set (learning sample), finding the best split, and then splitting again the resulting subspaces. The process is repeated by node number on all nodes which are not end-nodes (see figure 5.2 for node numeration scheme).

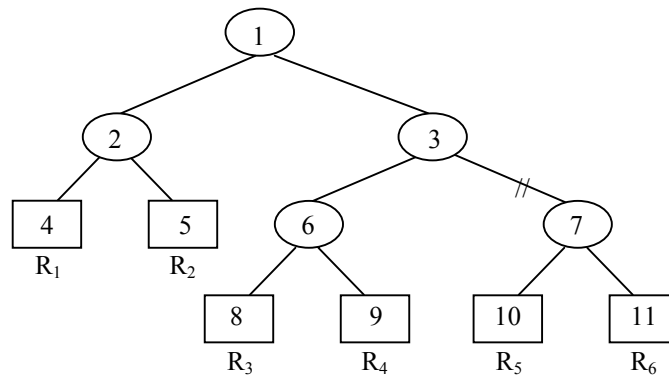


Figure 5.2: *Node numbering of trees*

5.1.2 Stopping criteria

The tree growing algorithm also needs to recognize when a node is final and when further splitting is needed. There are two stopping rules, which are used simultaneously. The current node is considered end node if:

1. A minimum, predefined amount of data points in it is reached.
2. No more splitting can be done, i.e. all data points in the node are described by the same vector of predictors (this is possible only if all predictors are categorical).

In addition, goodness of split criteria can be used to decide if a split is significant, or if the current node should be final. However, Hastie, Tibshirani, and Friedman (2001) note that this strategy is rather "short-sighted" since a "seemingly worthless split might lead to a very good split below it."

5.1.3 Pruning

It is preferable to grow a large tree first, knowing that it overfits the data, and then reducing it to the right size with the help of cost-complexity pruning for regression trees.

Let T_i be a subtree of the overgrown tree T_0 ; T_i is obtained from T_0 by pruning. Let $|T_i|$ denote the number of endnodes in T_i and let them be indexed by $m = 1, \dots, |T_i|$, corresponding to the regions R_m into which T_i splits the initial space (fig. 5.2). Then the cost-complexity criterion (also called error-complexity for regression trees) is defined to be (Breiman et al, 1984, Hastie, Tibshirani & Friedman, 2001):

$$C_\alpha(T_i) = \sum_{m=1}^{|T_i|} \sum_{x_i \in R_m} (y_i - \hat{c}_m)^2 + \alpha \cdot |T_i|$$

The cost-complexity criterion is an AIC type criterion, which penalizes by increasing the complexity of big trees. The tuning parameter $\alpha \geq 0$ governs the trade off between complexity and goodness of fit. The cost-complexity criterion is calculated in Appendix A at each possible pruning point for the tree in figure 5.2. Pruning is applied first at the weakest link in the tree model, which is followed by the node producing the smallest increase (per node) in the error term of the cost-complexity criterion. The result of pruning with parameter α would then be a tree $T_\alpha \subseteq T_0$, which minimizes the cost-complexity criterion. Notice that large α values result in smaller T_α trees and vice versa. If $\alpha = 0$, then $T_\alpha = T_0$.

Breiman et al (1984) describe the pruning theory in full length, and prove that there is a unique solution to the minimization problem.

5.2 Tree Performance

The tree model has lately become popular among medical scientists "... perhaps because it mimics the way that a doctor thinks." (Hastie, Tibshirani & Friedman, 2001). The main advantage of recursive binary partitioning is that the resulting models are interpretable. The whole space is partitioned in disjoint regions and all of them are described with a single tree.

The main disadvantage of tree models is their instability, which is due to the hierarchical nature of the model construction method. Bagging, for example, is a method developed to cope with this disadvantage (Breiman, 1996). Bagging averages among several tree models in order to reduce the variance, but the averaging process deprives the model from its main advantage – interpretability.

5.3 Responder identification with regression trees

As mentioned earlier, the responder identification idea is based on finding patients in the new treatment group, who are badly predicted by the prognostic model. In chapter 3 we have shown why martingale and deviance residuals are suitable for this purpose. Since regression tree models split the input space into regions, which are described by a part of the input variables (the predictive factors), and the size of the output variable in each region is predicted, we can use regression trees for responder identification. The tree would be built on the new treatment arm of the data and the residuals to the prognostic model would be used as an output variable. Binary splitting is based on maximal difference of the output variable in the regions, so the hope is, that one or more of the final regions of the tree model would have much larger or much smaller mean of the residuals in them, than the average for the input space (≈ 0).

The responder identification method is described in detail in chapter 8.

6. PRIM – PATIENT RULE INDUCTION METHOD (BUMP HUNTING)

Bump hunting, presented in this chapter as described by Friedman & Fisher (1999), is a Data Mining technique which optimizes a certain target function in order to find regions in the input space with special unique properties, different from the rest of the space. Unlike prediction models (like CART), which cover the whole input space and attempt to capture its general characteristics, bump hunting is designed to identify and describe only parts of the space, which gather elements with common special behavior.

Having a learning sample $L = \{y_i, x_i\}$, $i = 1, \dots, N$, $x_i = (x_{1i}, x_{2i}, \dots, x_{ni})$ taken from the underlying distribution with probability density of y at each x : $p(y|x)$, described by its first moment $f(x) = E(y|x)$, one can use the minimizer of the mean squared prediction error at each input x as a way of describing the underlying distribution with the help of the learning sample.

As previously mentioned, bump hunting is not interested in describing all of f . It merely searches for special properties of f , namely, its minima and maxima. This is a typical function optimization problem, which is solved by searching for regions of the input space (L) in which the average response values (y_i) are much larger (or much smaller) than the overall response average. Notice, that $\min f(x) = \max [-f(x)]$, so that the maximization algorithm can be transformed into a minimization one with just a simple sign change. For the rest of this chapter we will only discuss maximization without loss of generality.

Assuming that S_j is the set of all possible values of variable x_j , we can represent the entire input space as an n -dimensional product space:

$$S = S_1 \times S_2 \times \dots \times S_n$$

If we are not looking for a single maximal point x , but for a region containing that point, the goal of function optimization would be to find a subregion B of S , such that:

$$\bar{f}_B = \text{ave}_{x \in B} f(x) \gg \bar{f}_S.$$

The size, also called **support**, of region B is then the integrated probability density:

$$\beta_B = \int_{x \in B} p(x) dx.$$

Usually there is a trade-off between \bar{f}_B and β_B – larger function average is associated with smaller support in the region. Since we do not know the underlying distribution, but just have a learning sample, we will use the estimates of \bar{f}_B and β_B :

$$\hat{\beta}_B = \frac{1}{N} \sum_{x_i \in B} I(x_i \in B)$$

$$\bar{y}_B = \frac{1}{N \cdot \hat{\beta}_B} \sum_{x_i \in B} y_i$$

where y_i are the output values and function I is an indicator which (in the sum) counts the number of observations in region B .

Optimization theory offers many different strategies for function optimization. Bump hunting is a type of greedy algorithm equipped with patience, which stresses interpretability of the resulting regions.

6.1 General structure of the PRIM model

In general, bump hunting focuses on solutions which can be described in terms of important characteristics of the data. In particular, that means that the sought region B (also called **bump**) can be described by simple statements involving the input variables.

The final bump can be composed of several sub-regions (called **boxes**) B_k , $k = 1, \dots, K$, where B is the union of all sub-regions:

$$B = \bigcup_{k=1}^K B_k .$$

Each sub-region B_k is constructed of simple logical rules (called **borders**) involving different input variables. These borders are obtained by combining a certain number of basic rules that concern only one variable at a time. This makes the resulting region interpretable:

$$B_k = s_{1k} \times s_{2k} \times \dots \times s_{nk} ,$$

where s_{jk} is a subset of all possible values of x_j , $S_j: \{s_{jk} \subseteq S_j\}_1^n$. Thus, the sub-regions B_k are defined by the intersection of the subsets of all possible values of each single variable:

$$x \in B_k = \bigcap_{j=1}^n (x_j \in s_{jk}) .$$

If input variable x_j is continuous, subset s_{jk} will be an interval; if variable x_j is categorical, s_{jk} is a finite set of values from S_j .

Now we know that bumps are unions of boxes and boxes are intersections of borders. Next we need to know how to construct these elements. The construction of bumps happens stepwise. The calculation of the first box B_1 is done with the entire data set. Then the elements contained in B_1 are removed from the construction data set and the second box is constructed using the reduced data set. The k^{th} box is found using all data not included in any of the previous boxes. In this manner, since the successive box definitions depend on the previously constructed boxes, box-definitions grow more and more complicated, but all building blocks stay simple logical rules. This process of removing the constructed boxes, and thus, indirectly including them in the definition of all following boxes is called **covering**.

Theoretically, the entire input space can be covered with boxes, however this is not done, since, as mentioned earlier, bump hunting is designed for the identification of extreme regions of the target variable, rather than for an optimal approximation or a good prediction of the output over the entire input space.

6.2 Box construction

During the box construction process, just as in regression trees, bump hunting (PRIM) looks for rectangular regions (boxes), but not by minimizing the sum of the averages of the (two) new regions into which the current space is split. Bump hunting "peels off" a certain percentage of the data while optimizing the response average of the elements left in the box (see figure 6.1 for illustration of the box construction process with two input variables). At each peeling step, a variable and a peel-off value is chosen, which together define a border (the lines in figure 6.1) so that the data points left in the region have the largest mean of the output variable:

$$\max_{x_j, t_j} \bar{y}_B,$$

where B is the box resulting from a peeling at variable x_j and peeling point t_j . The **top-down-peeling** process stops when a minimum number of elements in the box is reached.

Since peeling is a greedy process, the average of the response variable in the box can often be improved by "pasting" back some of the data to the box (the checked regions in figure 6.1). The **bottom-up-pasting** process stops when the average in the box can no longer be improved. In general, when pasting is possible, the new box does have a larger mean of the response variable, but that rarely has a dramatic effect (Friedman & Fisher, 1999)

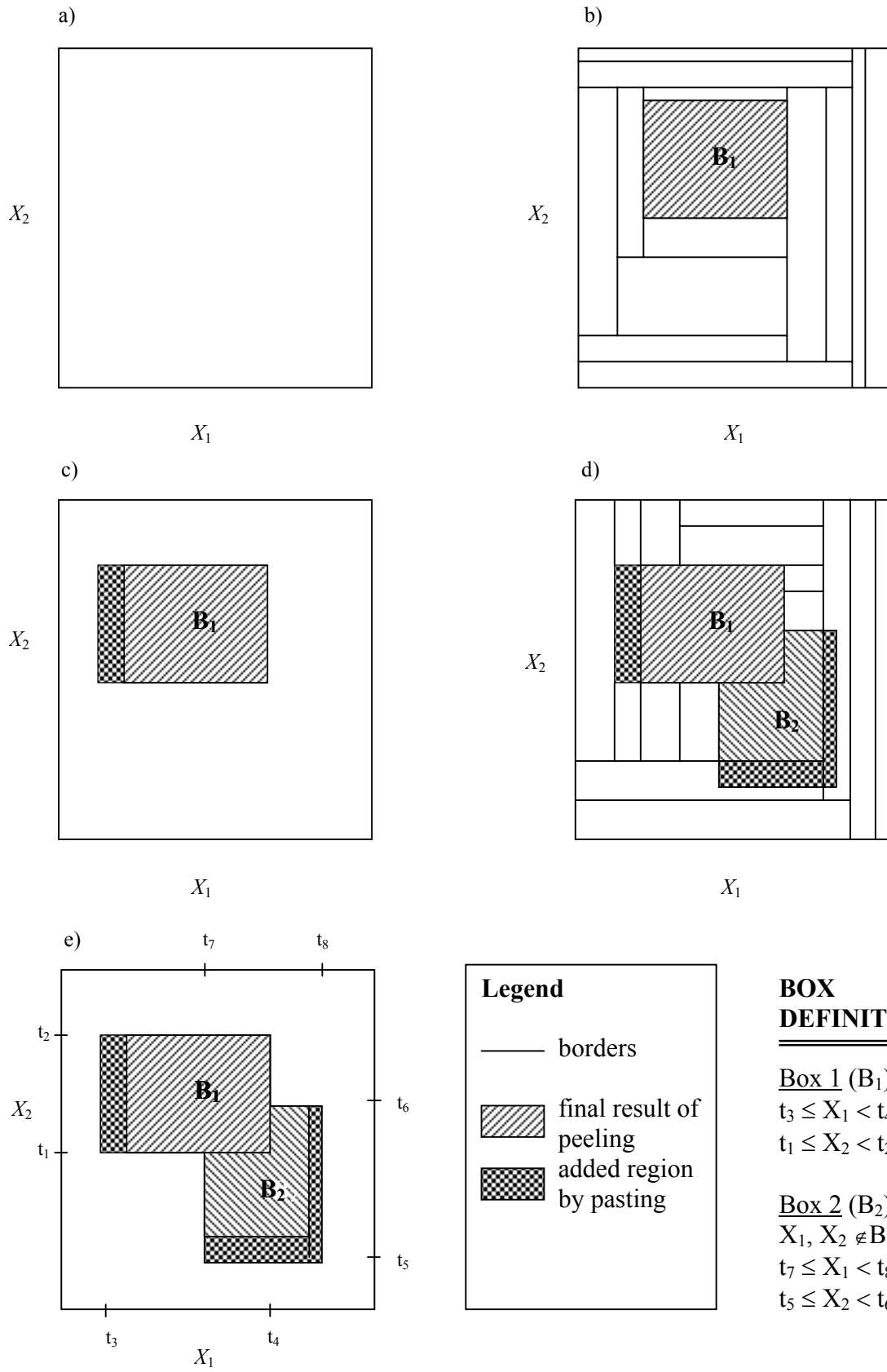


Figure 6.1: *Example of bump hunting model growth in a two dimensional input space (X_1 & X_2 are the input factors):*

- a) the entire input space*
- b) construction of the first box through peeling*
- c) improving the first box by pasting*
- d) construction of the second box through peeling in the input space, excluding the first box; improving the second box by two step pasting*
- e) the input space and the bump, consisting of two boxes; all borders are shown with factor and cut-value*

After covering, all resulting boxes together describe the maximal region – the resulting bump. Notice, that since each box is removed from the space before the construction of the next one, the following boxes are not always rectangular and, therefore, cannot always be expressed by a tree model.

Two parameters need to be specified in the box construction algorithm: **peeling quantile α** and **minimal support β_0** . The peeling quantile determines the percentage of data points excluded (peeled away) from the current box at each peeling step. Friedman and Fisher suggest values of α between 0.05 and 0.1, which results in the removal of 5% to 10% of the data at each step. The minimal support is a threshold parameter, which determines the minimal size of the final box. The choice of the minimal box support involves statistical and domain of application dependent considerations. The development of a box mean (i.e. mean of the target variable for data points in the box) with respect to support β can be observed with the help of the box construction **trajectory**. The trajectory allows one to visually choose an optimal β_0 . Figure 6.2 shows an example of a trajectory, constructed with $\alpha = 0.1$ where the mean of the response variable is maximized. One can observe how the mean grows from the mean of the whole data set (0) to about 0.75. The points on the trajectory represent the consecutively chosen borders. The trade off between support and mean in the growing box is clearly visible. Notice, maximization is done only in the direction of mean response. Multivariate optimization is not performed

If the input factors are categorical, that limits the peeling procedure in the following manner:

- Binary variables allow of only one peeling point, which splits the data into two parts.

- Variables with more than two categories are treated the same way as in CART. The peeling points are defined in such manner, that any category can be peeled off. Unlike continuous variables, where only the largest or the smallest values can be peeled off, in a categorical variable with, say, three categories: A, B, and C, category B can be peeled off (categories are not ordered).
- Continuous variables are often categorized. In this case, to preserve the order among the categories, we suggest artificially entering the categorized input variable as "continuous" in the bump hunting algorithm.

The general bump hunting algorithm delivers rather unstable models (just as CART does). If data permits, one can use cross validation at each box in order to stabilize the resulting model. If the initial space is small and does not allow cross validation, another stabilizing improvement of bump hunting can be used (see chapter 7).

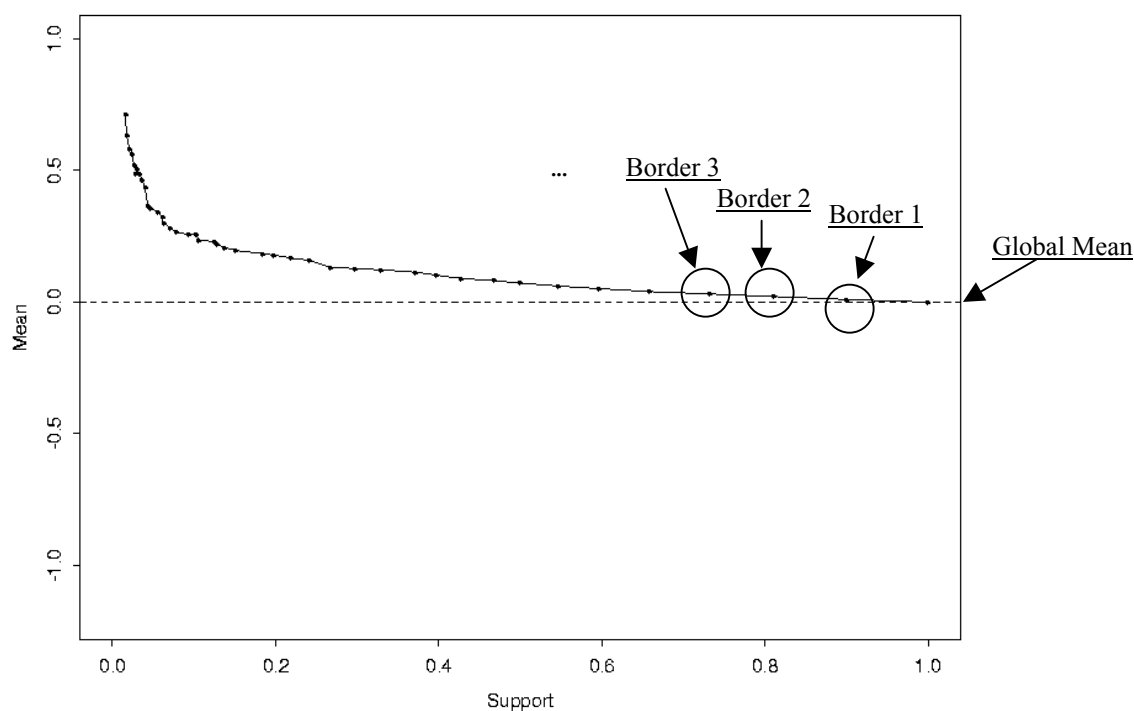


Figure 6.2: *Trajectory – visualization of the box-building process ($\alpha = 0.1$).*

A comparison between a tree and a bump model can only be made in a very loose sense, although both models partition the input space in regions, since trees describe the entire space and bumps – just extreme parts of it. A general advantage of bump hunting over CART is its patience. Due to the binary splitting in CART, the input space is quickly fragmented into large regions. Bump hunting peels off a certain adjustable proportion of data points at each step and can perform many more steps (on continuous factors) before running out of data.

For further detail on the bump hunting procedure, please refer to Friedman & Fisher (1999).

6.3 Responder identification with bump hunting

Although not created originally for responder identification, bump hunting seems to be tailor made for that purpose. Consider the function plotted in figure 6.3 to be the residual of a prognostic model of an imaginary (for visualization purposes two dimensional) input data set. The positive and negative bumps are clearly visible. The negative bump consists of two boxes; the positive one of three boxes. Bump hunting is designed precisely to identify the regions in the (two dimensional) input space, shown as the projection, where minima and maxima of the output function occur (the darker and lighter shaded areas). Applied to the residuals of the prognostic model, this procedure would identify patients with specific properties, who are not well predicted by the prognostic model. Those groups of patients, described by values of the input variables, would be candidates for positive and negative responders.

For details on the responder identification method, please refer to chapter 8.

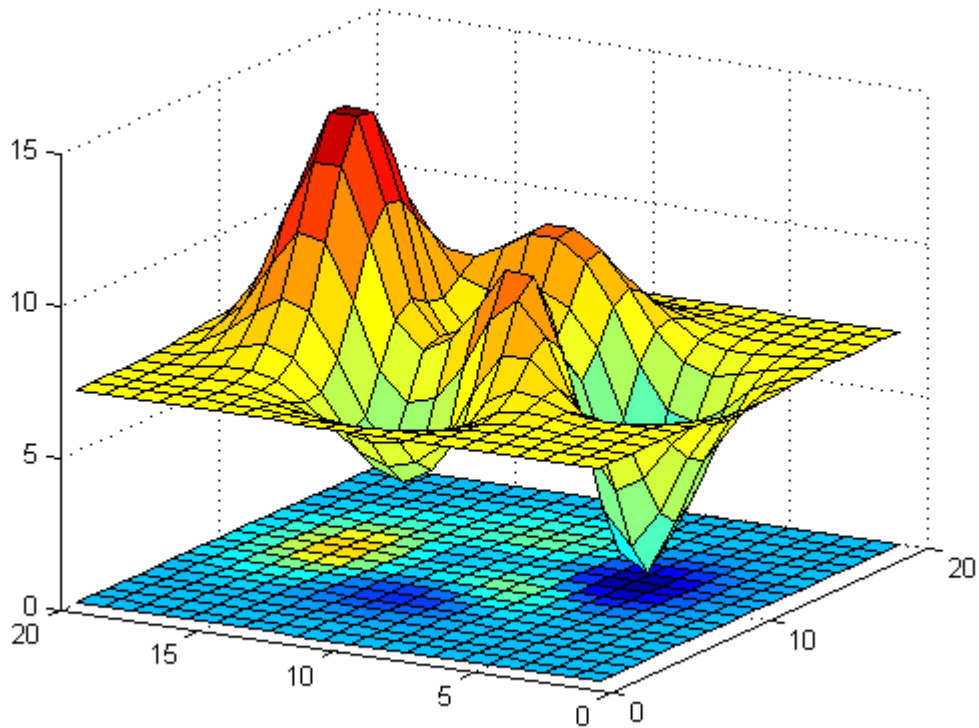


Figure 6.3: *Visualization of bump hunting for responder identification (the surface was created using function "peaks" in MATLAB 5). The horizontal coordinates represent a two dimensional input space, the vertical coordinate – the response variable. The two dimensional projection includes the response average on color scale, corresponding to each subdivision of the plane.*

7. STABILIZATION OF BUMP HUNTING

As mentioned earlier, bump models are rather unstable, due to their hierarchical nature. There are two general ways of assuring that a model is "good": validation and stabilization. Harrell, Lee, & Mark (1996) summarize the procedure for performing external validation and the types of internal validation: data splitting, cross-validation, and bootstrapping. The external and the first two internal validation methods require abundance of (appropriate) data, which one rarely has. Bootstrapping uses the entire data set in the model building process and then calculates some goodness-of-fit statistic on a large number of bootstrap samples taken from the original data.

There is no known goodness-of-fit statistic for the bump model. One can use the mean squared error as a type of homogeneity statistic, showing the difference between the mean of the output variable and the actual output values in the bump, but this does not capture the goodness-of-fit of the entire model. The choice of input variables and cut points is just as important. Therefore, bootstrapping as a validation procedure is not directly applicable for bump models. The validation choices left are external validation or internal validation involving data splitting. Both are not always possible.

If there is no direct way to validate a bump model, one should at least reduce the variability of the bump hunting model resulting from small changes in the data – the so called stabilizing.

7.1 Stabilizing with bootstrapping

Bootstrapping can be used as a stabilization procedure during the model building process (see Tibshirani & Knight, 1999, and Dannegger, 2000). One can use bootstrap samples of the original data in order to estimate the model coefficients or to choose

stronger predictors. Bootstrapping, in all of its shapes and forms improves or qualifies the predictive capacity of the model.

We apply bootstrapping in the method for identification of responders at each predictor selection step of the bump hunting process.

Referring back to chapter 6.2, figure 6.2, the borders involving different predictor variables are chosen one-by-one in the box-building procedure. Border 1 is represented in fig. 6.2 with its mean and support, i.e. using the restriction involving the predictor in Border 1, we obtain a box having a single border representing a set of patients from the treatment group. In order to choose the second border, the first one needs to be fixed. This hierarchical dependency leads to large variation in the bump hunting model after small alterations of the data set. We stabilize the bump hunting model in two ways:

1. Categorization of all continuous predictors needs to be done in order to reduce peel-off point variation. This limits somewhat the power of bump hunting, since it restricts the peeling process, but it is a necessary preliminary step for the bootstrapped bump hunting. We suggest using at least three categories, which can be either already known from previous studies cut points or the corresponding percentiles. Notice, however, that many of the predictors coming from the area of medicine can only be split into two logical parts (i.e. tumor size = increasing/decreasing), whereas others allow for more categories (i.e. age = child/adult/elderly). As suggested in chapter 6, such categorized continuous variables should be entered as "continuous" in the bump hunting procedure. See chapter 8.4 for a note on a more sophisticated procedure for cut point identification.
2. In order to stabilize the border selection method, we choose each border (i.e. predictor-restriction combination) after considering all borders chosen from n bootstrap samples. We fix a border and proceed to the next one only if it was chosen in the majority of the bootstrap samples.

The process in 2. is repeated according to the following algorithm:

7.2 Algorithm

1. Set the p -value of the log-rank statistic to 1 ($p(LR) = 1$) and let T be the set of all patients in the new treatment arm.
2. Take n bootstrap samples of T and, using the original bump hunting algorithm, create a trajectory for each one of them, including the original sample.
3. Consider all $n + 1$ first borders and the associated predictors and choose the one which appears most often. If there is a tie, choose the less restrictive border, i.e. one which results in a box with bigger support when applied to the original data set.

Note: Predictor and border are not equivalent terms. One predictor may appear with different restrictions in different bootstrap samples. We are only interested in the border frequency as a combination of predictor and constraint.

4. Restrict T using the border from step 3. Calculate the mean response and the support of the resulting box.
5. Apply the rules restricting T to the classical treatment (or placebo) group and create a set P of patients under the same restrictions as in T .
6. Calculate the log-rank statistic for the difference in survival between patients in P and in T . If $p(LR)$ improves¹ from its previous value, return to step 2. If not, stop.

Notice, that any stopping criteria which considers only one step at a time is easily implemented, but in general nearsighted. An alternative is to look several steps ahead before a stopping decision is made, since a seemingly "bad" border can lead to a "good" one and result in a better model (see section 10.5 for an example). We choose not to do this in the simulation study of chapter 9 in order to fully automate the software implementation and reduce computation time.

¹ The definition of "improves" can be different for different types of data. If initially there is no difference in survival between the new and the classical treatment groups, the p-value improves when it decreases. For cases where there is initial difference, please refer to chapter 8.2

As an example to the general algorithm above, the result of a single border selection step (steps 2 & 3) could be as summarized in table 7.1.

Table 7.1: *Hypothetical example of border selection with bootstrapping. AGE = 1 if patient age ≤ 30 years; AGE = 2 if patient age $\in (30, 65]$ years; AGE = 3 if patient age > 65 years. SMOKER = 0 if patient is non-smoker; SMOKER = 1 if patient is smoker.*

Restrictions:	AGE = 1	AGE = 2	AGE = 3	SMOKER = 0	SMOKER = 1
Original data	1	0	0	0	0
100 bootstrap samples	10	0	85	0	5

Table 7.1 should be interpreted as follows:

1. The first row shows all possible restrictions in the two input variables, i.e. all possible borders. Notice that the effect of restriction AGE = 1 is that all points with this characteristic would be chosen to be peeled off, i.e. AGE \neq 1 are left in the box.
2. The second row shows which border was chosen when ordinary bump hunting was applied to the original data.
3. The third row shows how many times each border was chosen by ordinary bump hunting in the 100 bootstrap samples of the original data.
4. Notice, that the ordinary bump hunting procedure would choose border AGE = 1 at this border selection step, whereas the stabilized procedure would choose border AGE = 3.

7.3 Discussion

The above algorithm can be modified to include pasting as well. In this case, we would use the minimal support (calculated in the original data set) as stopping criteria of the peeling process instead of the p -value of the log rank statistic. Pasting borders would also be chosen through bootstrapping. Here we can use both the p -value of the log rank statistic and the indicator for increase (decrease) of the box mean as stopping parameter.

Please refer to chapter 9 for implementation and comparison of performance between ordinary and stabilized bump hunting.

8. IDENTIFICATION OF RESPONDERS

Assuming that we have a good prognostic (Cox-PH) model, we can use bump hunting or regression trees for responder identification. One can use both martingale and deviance residuals of the prognostic model in order to identify predictive factors. Without loss of generality, we will concentrate on martingale residuals for the rest of this chapter. A systematic comparison of the performance of martingale and deviance residuals in responder identification is shown in chapter 9.

8.1 Algorithm using bump hunting

Consider the following strategy for positive and negative responder identification using a predictive model based on bump hunting (original or improved algorithm):

1. Develop a good prognostic Cox-PH model on the classical treatment arm of the data.
2. Apply the prognostic model together with its estimated coefficients and baseline hazard to the new treatment group.
3. Calculate the martingale residuals of the prognostic model in the new treatment group. Patients who are not well predicted (outliers in the residuals) would be candidates for responders.
4. Develop a bump hunting model on the new treatment group, using martingale residuals of the prognostic model as response.

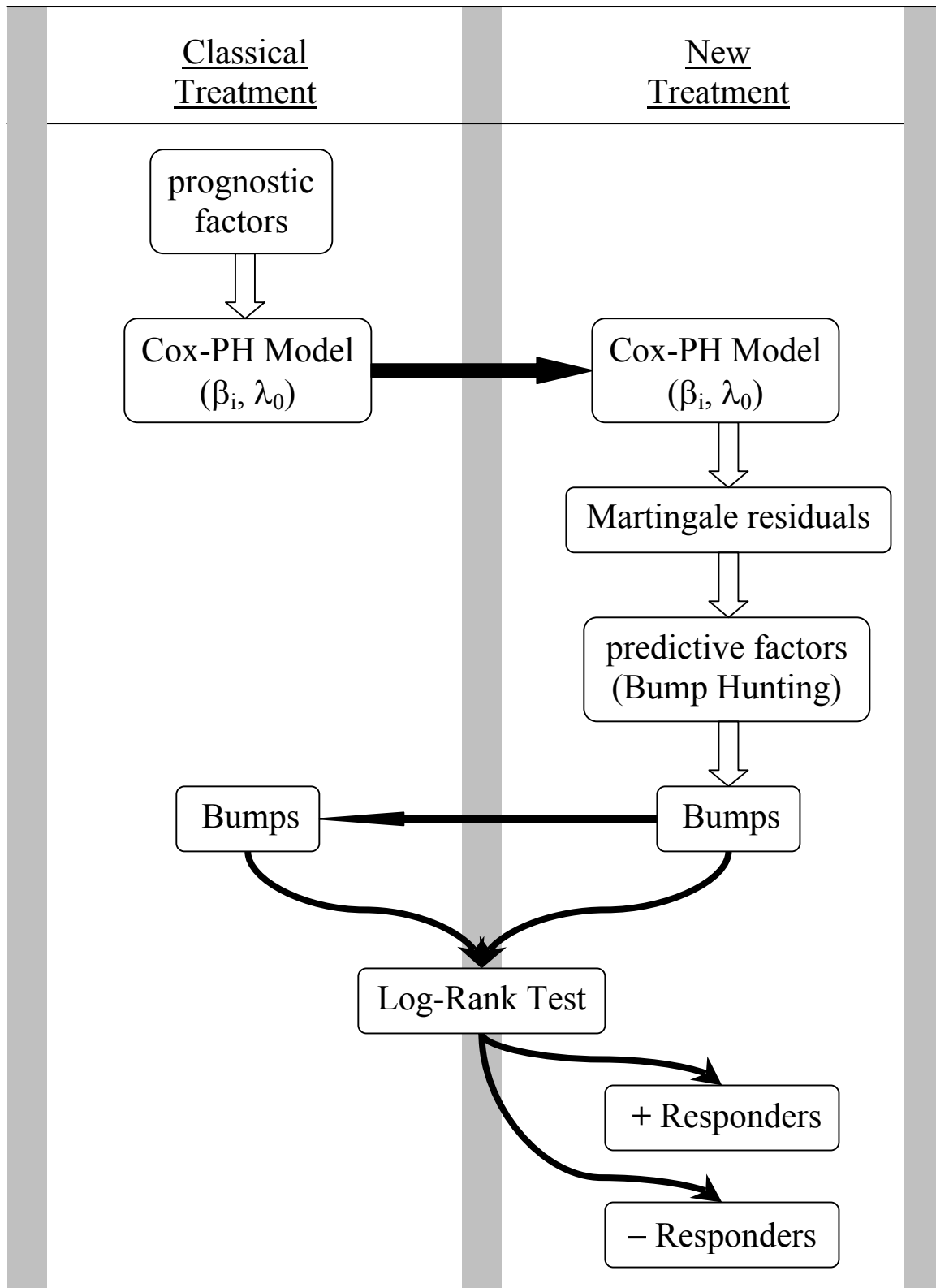


Figure 8.1. *Flow diagram of the responder identification algorithm.*

5. Identify the groups of patients in the classical treatment group, who correspond to the extreme groups (bumps) in the new treatment arm, i.e. divide the classical treatment space in the same way as the new treatment space and consider the regions which were identified as extreme (bumps) in the new treatment space. Compare the survival curves of each classical-new treatment pair of extreme regions (log rank test). **If there is a significant difference in survival, the group with extreme positive residuals would identify negative responders and the extreme negative residuals – positive responders.** Also, the factors involved in the description of the regions will be predictive. For illustration, please refer to figure 6.3.

The responder identification algorithm is shown schematically in the flow-chart of figure 8.1. The algorithm is tested in a simulation study (chapter 9). An application on the EMIAT data set can be found in chapter 10.

8.2 A note on survival curve differences

The responder identification method was developed with a situation in mind, in which overall the new treatment does not show to be better or worse than the classical treatment (i.e. the survival curves in both treatment arms do not differ significantly). The method needs slight alteration if initial difference in survival is at hand. Please refer to figure 8.2 for the following discussion.

In chapter 2 we have defined positive and negative responders starting out with two treatment arms which do not differ in survival as in case A (figure 8.2). Then a subgroup of positive responders is one, for which the new treatment increases significantly the survival rate of patients taking it, as in case B (figure 8.2). The negative responders on the other hand are such patients in the new treatment group, who's survival time is significantly shorter than that of an equivalent group of patients taking the classical (old) treatment, as in case C (figure 8.2).

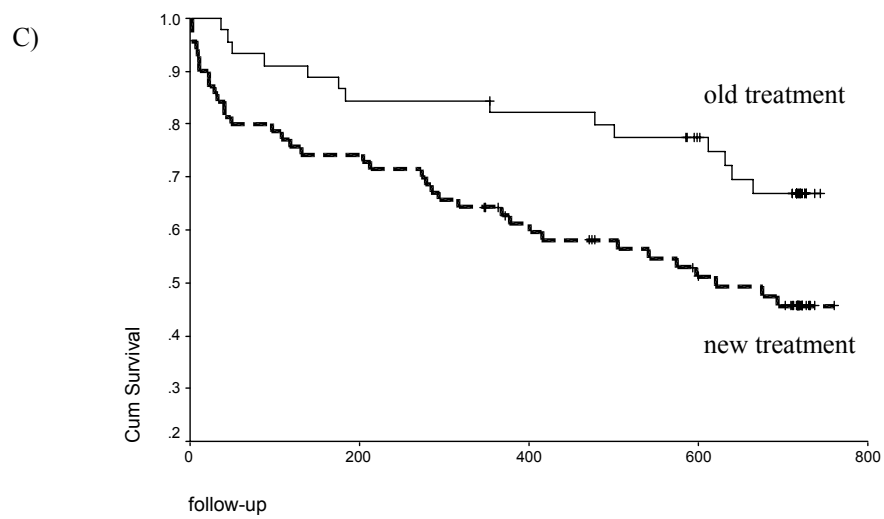
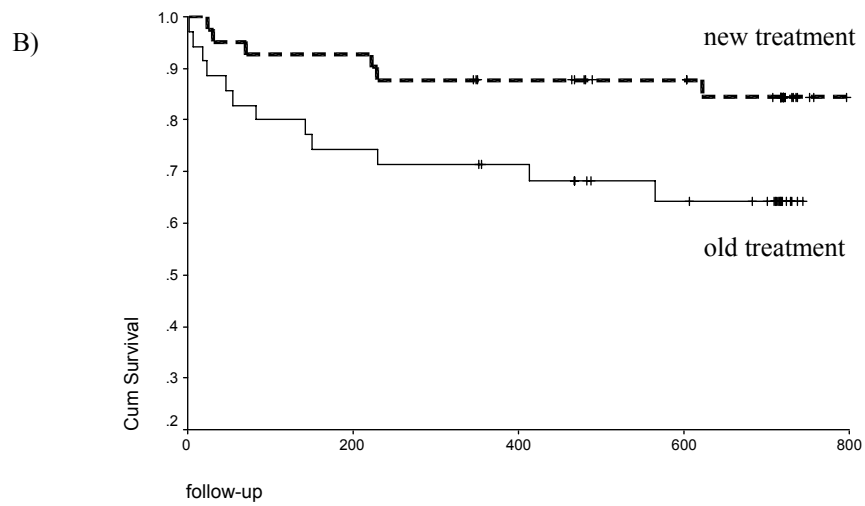
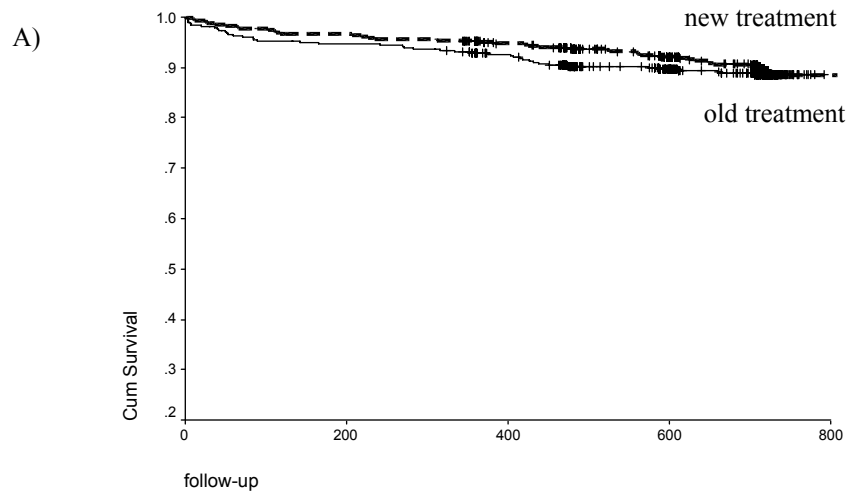


Figure 8.2: *Possible overall survival curves in a clinical trial with two arms:*
 A) *No difference between the survival curves of the two arms*
 B) *The new treatment is overall better than the old one*
 C) *The new treatment is overall worse than the old one*

Let us consider now a clinical trial, in which the new treatment is better than the old one, when the entire patient populations are compared (case B). Then the entire new treatment group would be considered positive responders. In this case, it would be interesting to know if a certain subgroup of patients under the new treatment are actually harmed by it (as in case C). They would be considered negative responders. There might be a subgroup of patients taking the new treatment, who do just as well as their counterparts taking the classical treatment (case A). In this case, those patients would not be considered to be negative responders, since they are not harmed by the new treatment, although they do not profit from it either.

Unfortunately, there are also trials in which the new treatment patients show worse survival rates than the classical treatment patients (case C). In such disastrous trials one would be interested to know if a certain group of patients taking the new treatment actually do survive longer than a similar group taking the old treatment (as in case B). This would be a group of positive responders among the entire population of negative responders. Just as in the previously described situation, case A would be of no interest.

In the last two trial scenarios, the algorithm for responder identification needs slight alteration. Ordinarily, one would use the change in p-value of the log-rank statistic as a stopping criteria in the bump hunting procedure (chapter 7.2, point 6 and chapter 8.1, point 5). In the peeling process of bump hunting, one would reduce the new treatment patient arm step by step. If there is a significant difference (case B or C) for the entire population, reducing the group would lead to less and less significant p-values before it eventually reduces the new treatment group to this one special subgroup, for which the p-values become significant again to show difference in survival between the new and the old treatment subgroups in the opposite direction from the initial situation (as in case C or B). In those cases, the p-value of the log-rank statistic cannot be used as an automatic stopping criteria and the growth of the bump hunting model needs to be

controlled manually. Alternatively, the algorithm can be changed to "intelligently" evaluate the p-values by looking some steps ahead in the algorithm.

8.3 Changes to the responder identification algorithm if CART is used

It is also possible to use a regression tree instead of a bump model. In this case, one would construct a regression tree model in step 4 of the algorithm in 8.1 and change the algorithm from that point on as follows:

Steps 1 through 3 as in the algorithm in section 8.1.

4. Develop a regression tree model on the new treatment group, using the martingale residuals of the prognostic model as response.

Note: A tree model describes the whole input space. We are interested only in extreme regions, i.e. end nodes with patients who have large positive or large negative residuals (e.g. R_1 & R_3 and R_4 & R_6 in figure 5.1). Notice that it is quite possible to have tree models which do not identify extreme regions or just deliver a negative or just a positive one. This depends on the tree complexity and on the data at hand. If an extreme region, positive or negative, is identified, the factors involved in defining it will be candidates for predictors.

5. Order all end nodes by size of the mean response in them.
6. Split the classical treatment group into subgroups as defined by the end nodes of the regression tree model.
7. Start with the largest in absolute value (by mean of response) negative end node in the new treatment arm. Compare the survival curves of the new and classical treatment patients identified with that node. Calculate the p-value of the log-rank statistic.
8. Add the patients contained in the next largest negative end node (from the list in 5.) to the previously considered group of new treatment patients. Calculate p(LR) for the identified groups in the new and classical treatment arms.

9. Repeat step 8 while $p(\text{LR})$ decreases or until there are no more negative end nodes.
10. The last combination of end nodes defines the set of positive responders. The factors involved in defining it are predictive.
11. Repeat steps 7 through 9 for the non-negative end nodes from the list in 5., starting with the largest (by mean of response).
12. The last combination of non-negative end nodes defines negative responders. The factors involved in defining it are predictive.

Notice again, that the algorithm can be improved, if it looks at p -values several steps ahead before it stops the growth of the responder groups.

This strategy is tested in the simulation study of chapter 9 and applied on the EMIAT data set in chapter 10.

8.4 Covariate considerations

Obviously, continuous predictors give a large choice of borders (split-points in CART). Limiting this choice by categorizing continuous variables stabilizes the resulting model (see chapter 7.1). However, even though we recommend and perform categorization before bump hunting, one needs to carefully consider the dangers of such a procedure (see Altman et al, 1994).

The most simple and most often applied categorization procedure is splitting at the median when dichotomizing or at the appropriate percentile for more than two categories. This results in equally sized categories, but the cut points are generally not "optimal." For some well established prognostic factors one could take the cutpoints which were used in previous studies. This is also dangerous, especially in cases, in which the new study differs in some major point from the old one (which is usually the case). Altman et al (1994) have suggested the minimal p -value approach instead – a more sophisticated categorization technique, which exists in different variations since 1994.

9. SIMULATION STUDY

The responder identification method proposed in chapter 8 requires justification. If there is a data set, in which the positive and negative responder groups are known, one can apply the different versions of the algorithm discussed up to now and compare their ability to recognize those groups. Unfortunately, this is not possible in a real life data set, since the actual groups to be identified are not known. On the other hand, one can simulate survival data, in which the positive and negative responders are known. The full analysis and comparison of the responder identification algorithm requires a carefully planned simulation study.

9.1 Methods

A survival type data set was simulated to resemble a two arm randomized clinical trial with a total of 1000 patients, in which no difference in survival is observed between the two treatment groups (some structure of the EMIAT data set was mimicked). A total of seven factors were created: five binomial, one categorical with three levels, and one continuous in order to test the power of the different procedures in dealing with different types of variables. The factors were simulated in the following way:

Binary factors $X1$, $X4$, $X5$, $X6$, and $TREAT$:

Each factor is a vector of length 1000, each component of which is chosen at random from a binomial distribution with probability $p = .5$. $TREAT = 0$ denotes placebo patients, $TREAT = 1$ denotes new treatment patients.

Categorical factor $X2$:

Factor $X2$ is a vector of length 1000, the components of which were chosen at random from the set $\{0, 1, 2\}$ with corresponding probabilities $\{.33, .33, .34\}$.

Continuous factor X3:

Factor $X3$ is a vector of length 1000 with components chosen at random from a normal distribution with mean five and variance two.

Follow-up time for the data set was simulated in a way, which assures that the following Cox-PH model with prognostic and predictive parts would fit the data set:

$$\lambda(t | X) = \lambda_0(t) \cdot e^{\overbrace{\beta_1 \cdot X1 + \beta_2 \cdot X1 \cdot X3 + \beta_3 \cdot X3}^{\text{prognostic}} + \overbrace{c_{\min} \cdot X4 \cdot X5 \cdot X6 \cdot TREAT + c_{\max} \cdot X22 \cdot X5 \cdot TREAT}^{\text{predictive}}} \quad (9.1)$$

where:

X is the matrix of factors in the model

c_{\min} and c_{\max} are coefficients in the predictive part

$$X22 = \begin{cases} 1 & \text{if } X2 = 2 \\ 0 & \text{else} \end{cases}$$

β_1 , β_2 , and β_3 are coefficients in the prognostic part

The following pairs of values for c_{\min} and c_{\max} were chosen for further investigation: (-2, 2), (-1, 1), and (-.5, .25). Note, that larger absolute values of the coefficients simulate stronger influence of the predictive part of the model on the hazard. In addition, since c_{\min} is always negative, this term of the model decreases the hazard, i.e. patients with $X4 = 1$ & $X5 = 1$ & $X6 = 1$ would have lower hazard under treatment than under placebo – this is the simulated positive responder group. Conversely, since c_{\max} is always positive, that term would increase the hazard, i.e., patients with $X2 = 2$ & $X5 = 1$ would have higher hazard under treatment than under placebo – this is the negative responder group.

The values for the prognostic coefficients $\beta_1 = \ln 3$, $\beta_2 = -(\ln 3)/5$, and $\beta_3 = (\ln 3)/10$ (≈ 1.0986 , $-.2197$, and $.1099$ respectively) simulate an interaction between categorical factor $X1$ and continuous factor $X3$ as depicted in the relative hazard plot of figure 9.1. The interaction can be interpreted the following way: in absence of factor $X1$ ($X1 = 0$), increase of factor $X3$ (from 0 to 10) increases the relative hazard (from 1 to 3); in presence of factor $X1$ ($X1 = 1$), increase of factor $X3$ (from 0 to 10) decreases the relative hazard (from 3 to 1).

Follow-up time for this model was simulated to be Weibull distributed with shape parameter equal to two and scale parameter equal to the relative hazard, i.e. TIME was created to be a vector of length 1000, each component of which was chosen at random from the unique to each patient Weibull distribution, depending on his/her relative hazard ($= \lambda(t|x)/\lambda_0(t)$).

Censoring was simulated in the usual way by first defining a temporary vector TEMP of length 1000, containing random values of the uniform distribution on the interval $[0, \tau]$. Vector DEATH indicating event was then defined to be:

$$\text{DEATH}_i = \begin{cases} 1 & \text{if } t_i \leq \text{TEMP}_i \\ 0 & \text{else} \end{cases}$$

This procedure for assigning censoring allows for regulation of the percent censored cases in the data through parameter τ and assures, that censoring is assigned independently of time. Three different values of τ were considered: (11, 2.15, .65), which result in three different percentages censoring in the data: \approx (10, 30, 70)% respectively.

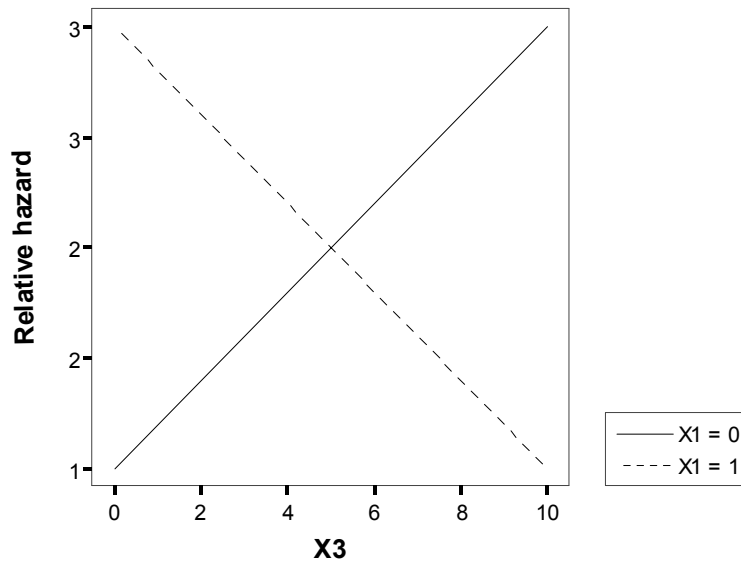


Figure 9.1: *Interaction between X1 & X3*

The above described survival data set was simulated a in groups with different model coefficients and censoring rates as shown in table 9.1.

Table 9.1: *Simulation groups*

Simulation group #	C_{min}	C_{max}	τ
1	-2	2	11
2	-1	1	11
3	-.5	.25	11
4	-2	2	2.15
5	-1	1	2.15
6	-.5	.25	2.15
7	-2	2	.65
8	-1	1	.65
9	-.5	.25	.65

The survival curves of the placebo (TREAT = 0) and treatment (TREAT = 1) groups were compared in each simulation group. The difference was not significant at the .05 level as it was expected by the simulation study design. Scatter plots of martingale residuals vs. follow-up time as well as deviance residuals vs. follow-up time are given in appendix B for the placebo and treatment groups of one data set in each of the nine simulation groups. One can observe how the different percentage of censoring and values of the predictive coefficients influence the residuals.

The **actual positive responder** group in all simulations contained data points with TREAT = 1 and all of the following constraints:

$$X4 = 1 \ \& \ X5 = 1 \ \& \ X6 = 1$$

The **actual negative responder** group contained data points with TREAT = 1 and both constraints:

$$X2 = 2 \ \& \ X5 = 1$$

The simulated responder groups differ in survival between the treatment and placebo groups (as expected). For example, a data set from simulation group 2 would have p-values of the log-rank statistic, which are significant at the .001 level (please refer to figure 9.2 for Kaplan-Meier survival curves in the responder and non-responder groups

of $TREAT = 0$ and $TREAT = 1$). Figure 9.3 shows the martingale and the deviance residuals for both responder groups plotted against follow-up time in the placebo and treatment groups.

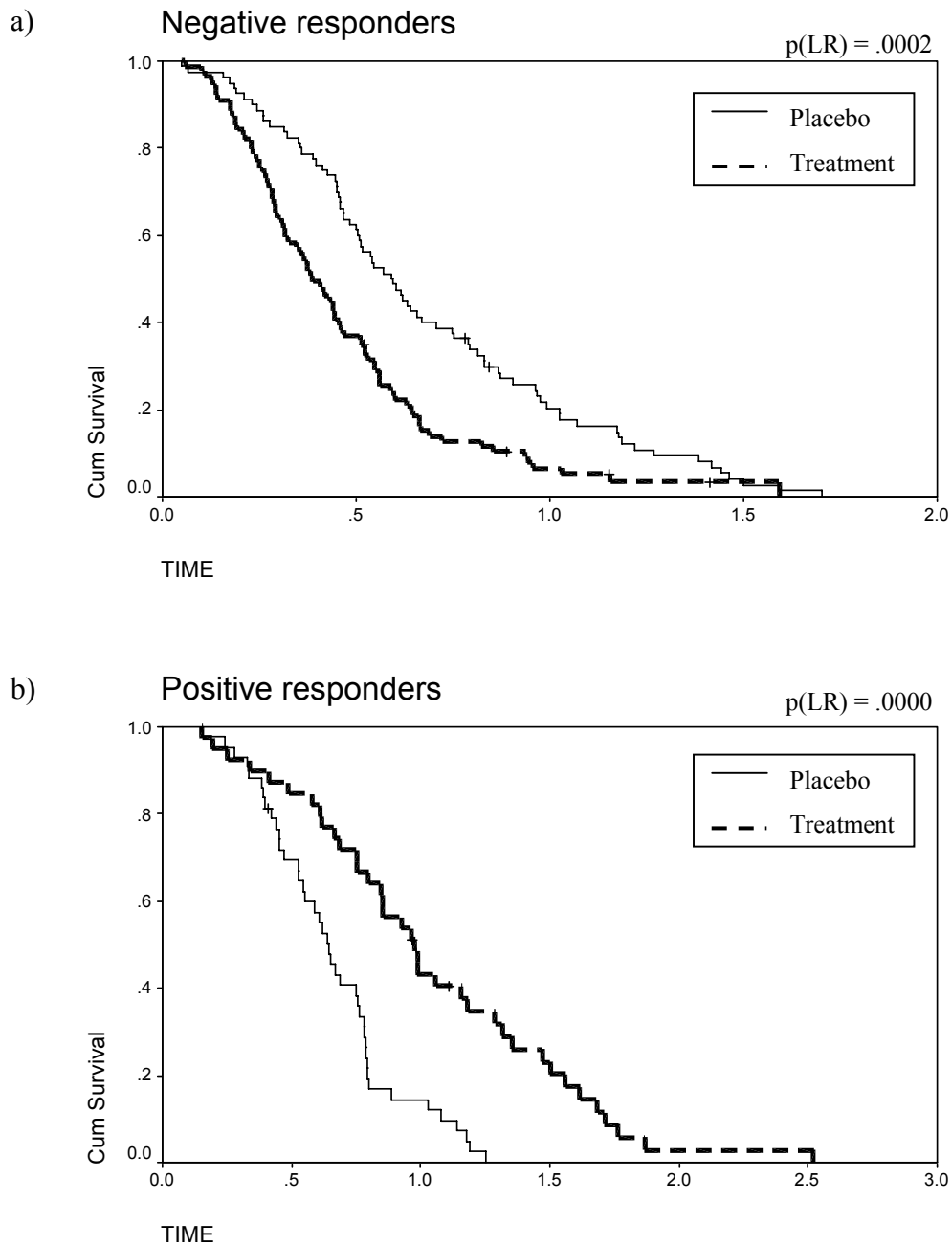


Figure 9.2: *Kaplan-Meier survival curve estimates in the actual positive and negative responder groups for a data set in simulation group 2.*

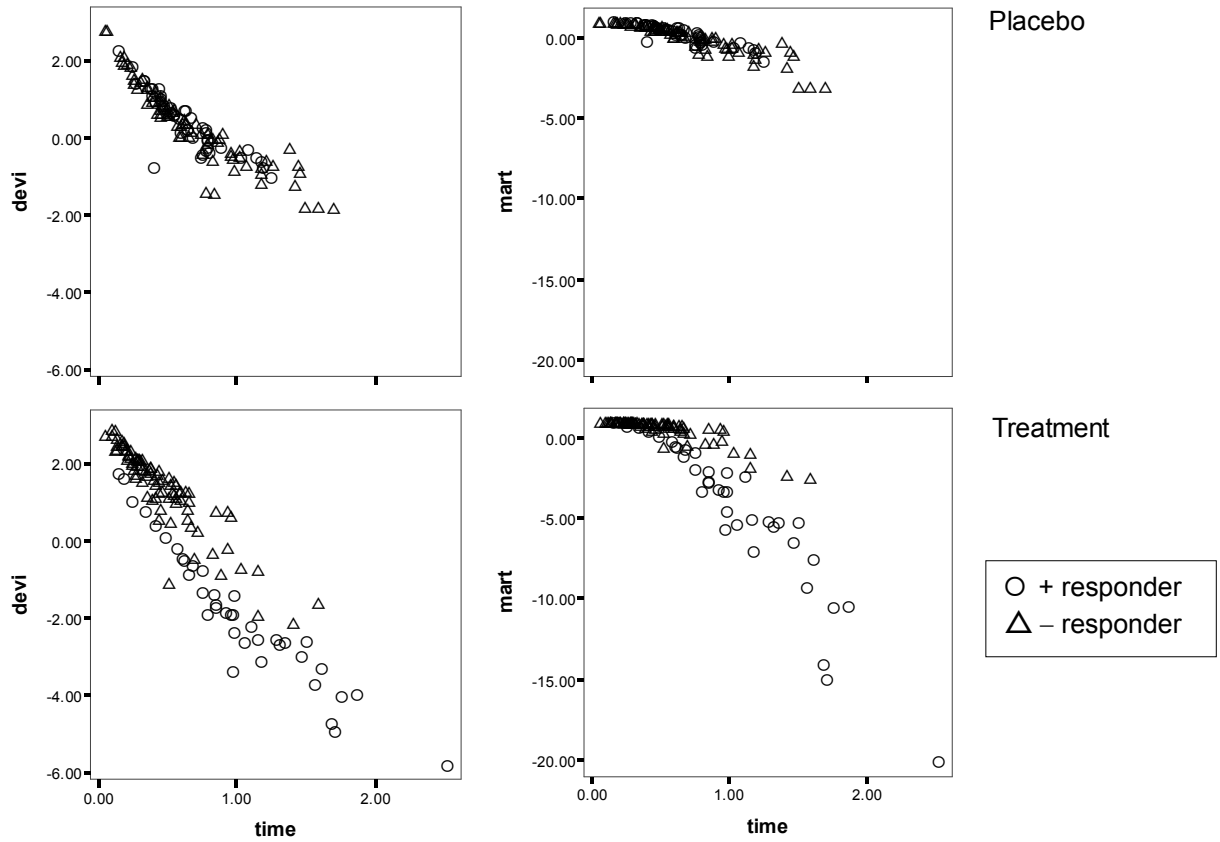


Figure 9.3: Scatter plots of the deviance and martingale residuals in the placebo and treatment groups against follow-up time for the positive and negative responder groups of figure 9.2.

9.2 Results

9.2.1 Cox-PH with interaction

The simulation study was designed in such a way, that a certain Cox-PH model with treatment interaction terms (model 9.1) should fit the data. We fit model 9.1 to all simulated data sets in order to check the simulation method. The mean of the Cox model coefficients over 100 data sets in each simulation group are shown in table 9.2. Notice, those coefficients are very close to the simulated ones, as they should be, so we can conclude that the data simulation was done correctly. The Wald statistic for factor

input to the model was significant in general at the .01 level, except for models constructed on data sets from simulation group 9. Apparently, the Cox-PH model cannot distinguish the simulated effect of the factors on hazard from noise when that effect is not strong (c_{\max} and c_{\min} are small in absolute value) and there is high percentage of censoring in the data.

Table 9.2: *Mean coefficients of model 9.1 over 100 simulations*

Simulation group #	β_1	β_2	β_3	c_{\min}	c_{\max}
1	1.1116	-.2234	.1116	-1.9925	2.0039
2	1.1205	-.2254	.1142	-.9981	.9942
3	1.1120	-.2233	.1105	-.5019	.2628
4	1.1238	-.2276	.1104	-1.9971	2.0226
5	1.0935	-.2193	.1100	-1.0000	1.0030
6	1.1433	-.2268	.1138	-.5048	.2629
7	1.0857	-.2156	.1074	-2.1266	2.0410
8	1.0970	-.2205	.1143	-1.0623	.9543
9	1.0607	-.2143	.1008	-.4950	.2402


Three representative simulation groups were chosen in an attempt to evaluate the power of the most frequently used variable selection process, forward stepwise selection, to identify the simulated Cox-PH model with interactions (model 9.1) as "best." Forward selection with likelihood ratio test as model improvement criteria was used with inclusion $p(\text{Wald}) = .01$ and exclusion $p(\text{Wald}) = .05$. Table 9.3 gives a summary of this investigation. Simulation groups 1, 5, and 9 were chosen as representative. Simulation group 1 has strong simulated treatment effect (easy to detect) and only 10% censoring. Simulation group 5 has medium strength simulated treatment effect and 30% censoring. Simulation group 9 has 70% censoring and slight treatment effect (difficult to detect). A total of 10 data sets were simulated in each group. The null model (no factors) and the correct model likelihood ratios were computed on each data set. Forward stepwise selection was applied on each data set four times: once including all factors X_1 through X_6 and TREAT and all their possible two-way interactions (a total of $7 + 21 = 28$ factors to choose from), once including all single factors and all their up to third order interactions ($28 + 35 = 63$ factors), all single factors and all their up to fourth order interactions ($63 + 35 = 98$ factors), and finally, all single factors and their up to fifth

order interactions ($98 + 21 = 119$ factors). The largest interaction term in the correct model is of fourth order. Interactions of up to fifth order were considered in order to check if forward selection including interaction terms of higher than needed order would choose more complicated terms than necessary. This should give a hint on the behavior of the automated selection procedure in a "real life" data set, for which the correct model is unknown.

Table 9.3: *Table of likelihood ratios for the null model, the models found with forward selection when different highest order interactions were present, and the correct model.*

Simulation group	Run	Null (df=0)	2 ^d order interaction (df)	3 ^d order interaction (df)	4 th order interaction (df)	5 th order interaction (df)	Correct (df=6)
1	1	11239	11076 (11)	10976 (8)	10942 (6)	10942 (6)	10942
	2	11110	10941 (9)	10860 (5)	10844 (4)	10844 (4)	10820
	3	11121	10986 (10)	10884 (9)	10860 (5)	10860 (5)	10814
	4	11161	10946 (9)	10893 (6)	10868 (4)	10857 (6)	10822
	5	11094	10966 (9)	10787 (11)	10803 (5)	10803 (5)	10764
	6	11093	10929 (10)	10861 (5)	10819 (5)	10804 (8)	10784
	7	11132	10914 (13)	10801 (12)	10736 (10)	10736 (10)	10761
	8	11103	10919 (11)	10857 (5)	10819 (5)	10819 (5)	10803
	9	11109	10998 (6)	10878 (7)	10828 (7)	10828 (7)	10805
	10	11189	11015 (11)	10912 (12)	10894 (8)	10894 (8)	10903
5	1	8651	8608 (3)	8593 (5)	8581 (5)	8581 (5)	8555
	2	8705	8677 (3)	8633 (6)	8636 (5)	8636 (5)	8632
	3	8441	8422 (2)	8394 (5)	8395 (3)	8382 (5)	8366
	4	8525	8491 (5)	8476 (3)	8472 (3)	8472 (3)	8447
	5	8407	8381 (3)	8321 (9)	8329 (7)	8329 (7)	8324
	6	8412	8387 (3)	8365 (5)	8363 (4)	8363 (4)	8343
	7	8444	8410 (3)	8401 (3)	8384 (4)	8384 (4)	8368
	8	8584	8529 (4)	8522 (3)	8503 (3)	8503 (3)	8472
	9	8428	8397 (3)	8342 (4)	8335 (4)	8335 (4)	8316
	10	8586	8547 (3)	8495 (3)	8486 (3)	8486 (3)	8453
9	1	2740		2722 (3)	2722 (3)	2722 (3)	2724
	2	2474					2455
	3	2614					2600
	4	2643		2627 (2)	2627 (2)	2627 (2)	2631
	5	2845	2829 (3)	2829 (3)	2829 (3)	2829 (3)	2823
	6	2956					2945
	7	2509					2489
	8	2738	2729 (1)	2729 (1)	2729 (1)	2729 (1)	2724
	9	2751					2745
	10	2291					2279

bold = models with better LR than the corresponding correct model

 = identical models (valid for the row)

In defense of the forward stepwise selection procedure, one should note, that in most cases (21 out of 24 constructed models) it did not add a fifth order interaction term, but delivered the model chosen from the procedure including up to fourth order interactions (see table 9.3). Unfortunately, it also chose the correct model only once out of 30 times.

Consider first simulation group 9, the most realistic one. When the correct model 9.1 was applied on the 10 data sets, it always reduced the likelihood ratio from the null model, but not always significantly (3 out of 10 were significant at the .01 level). In addition, the factor coefficients were also not significant (at the .01 level). In 6 out of 10 cases the forward selection procedure did not find any significant factors. In the 4 data sets, in which significant factors were found, they were other than the simulated ones (i.e. noise).

The overall impression is that the Cox-PH model with interactions is not a sensitive enough method for responder identification purposes when the effect of factors and factor combinations on treatment is weak and there is large percentage of censoring in the data.

In simulation groups 1 and 5 (see table 9.3), the likelihood ratio of the correct model was better than that of the forward selected models for 18 out of 20 data sets (notice, in the cases where $LR(\text{forward}) < LR(\text{correct})$ the forward selected model contained the correct model and some additional factors). The coefficients of the correct model were always significant at the .01 level and most often at the .001 level. The Cox-PH model with interactions performs well on data sets with small to moderate percent censoring and strong to moderate treatment effect. The problem with applying this responder identification procedure in praxis is that the correct model is unknown and the forward selection procedure has low power (it chose the correct model only once out of 20 times!).

9.2.2 Regression trees

The regression tree version of the responder identification method described in chapter 8 was applied on the following simulated data. In each of the nine simulation groups from table 9.1, 200 data sets were constructed as described in section 9.1. Martingale residuals were calculated on half of the data sets in each group. Deviance residuals were calculated on the other half.

Goal: Using the known positive and negative responder groups in each data set:

- 1) to evaluate the prognostic power of the responder identification procedure with regression trees (described in chapter 8), i.e. to compare the identified through the method groups of responders to the correct groups
- 2) to compare the power of identification of the method when martingale and deviance residuals are used as response variables in the regression tree model.

Step 1 of the responder identification algorithm can be skipped in the simulation study. The prognostic model here is known. It was simulated to contain factors X_1 , X_3 , and their linear interaction. This Cox-PH model was applied on the placebo part ($TREAT = 0$) of each data set, where the three model coefficients and the baseline hazard were estimated. As shown in 9.2.1, the estimated coefficients would be very close to the simulated ones. Using the so estimated model coefficients and baseline hazard, martingale (or deviance) residuals were calculated on the treatment ($TREAT = 1$) groups. The treatment groups of each simulated data set were used to construct regression tree models. All factors were included in the model selection procedure. Factor X_3 was used dichotomized at the mean and the residuals were used as response variable. All regression tree models were pruned to size 5 (five end nodes), which showed to be sufficiently large to include all known predictive factors. A typical resulting tree is shown in figure 9.4. Generally, the correct predictive factors were chosen, but not always in an optimal arrangement, which is crucial for the responder identification power of the model. The pruned tree models were "applied" on the placebo groups and the data was divided into subgroups, corresponding to the end

nodes. Each pruned tree couple was analyzed separately. All end nodes were arranged by size of the mean response in the treatment arm.

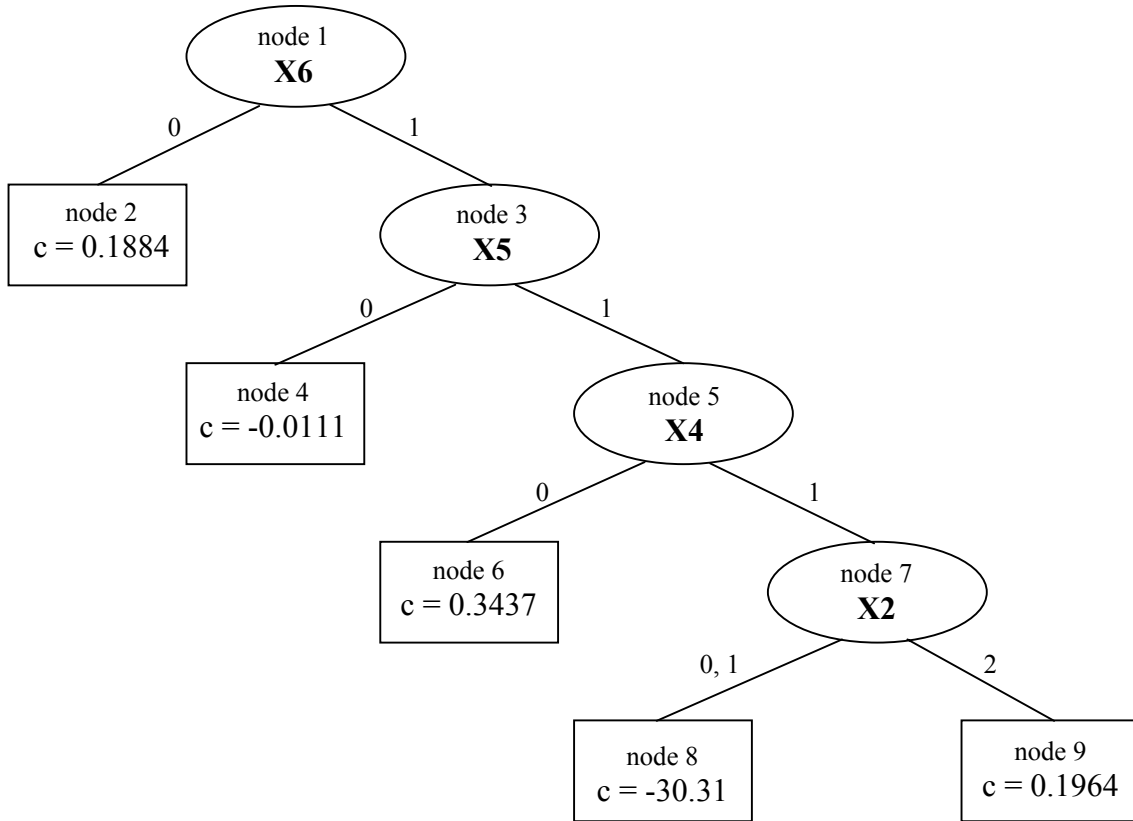


Figure 9.4: A sample regression tree built on a simulated data set, pruned to 5 end nodes (c = mean response).

Starting with the largest negative node, as described in the algorithm of section 8.3, we calculated the log-rank statistic for difference in survival of the placebo and treatment patients in the node. Then we made a joint group of patients, containing the two most negative nodes and calculated the log-rank statistic for this group. We kept adding negative end nodes to the group according to their size until there were no more negative end nodes or until the p-value of the log-rank statistic stopped improving. The group of chosen negative end nodes defined positive responders.

The same procedure was repeated for the positive end-nodes, starting with the largest one. The result was a set of negative responders in each simulation.

The so defined responder groups were then compared to the correct groups. Table 9.4 shows the average number of times the entire responder sets were chosen correctly over the 100 data sets in each simulation group and each residual type. Since this did not happen very often, the mean percentage of correctly identified patients was also calculated (see table 9.4).

Table 9.4: *Mean percent (%) correctly identified responders and number of times (#) the correct responder groups were chosen from 100 simulations for each simulation group, when regression tree is used as predictive model. MART = martingale residuals; DEVI = deviance residuals.*

sim. group #	MIN end nodes (+ responders)				MAX end nodes (– responders)			
	MART		DEVI		MART		DEVI	
	#	%	#	%	#	%	#	%
1	98	98	100	100	0	32.53	65	72.67
2	73	73	94	94	23	47.71	71	73.31
3	22	37.57	52	63.80	10	41.88	10	33.74
4	94	94	94	94	0	33.99	81	81.68
5	76	77.15	76	76	41	59.24	70	75.39
6	28	45.02	33	50.79	6	35.59	6	30.40
7	66	66	59	59	60	68.95	52	70.84
8	51	53.08	53	56.64	53	61.73	32	55.89
9	21	38.37	10	24	4	22.74	1	16.08

The results were better for data with low percentage censoring (sim. groups 1, 2, & 3 \approx 10%) than for data with high percentage censoring (7, 8, & 9 \approx 70%), except for simulations with large coefficients in the positive nodes (1 & 4) with martingale residuals. This might be explained by the fact that $MART \in (-\infty, 1]$, whereas $DEVI \in (-\infty, \infty)$. Large c_{\max} delivers large positive residuals, so that with 90 and 70 % of the data being in the interval $(0, 1]$, a lot of martingale residuals would be very close to 1. CART is based on splitting the data space into regions with most different mean of the response variable (here MART) and it obviously has a problem with simulation groups 1 & 4. Deviance residuals, on the other hand, "stretch" the positive side of the martingale residuals distribution, so that CART shows to perform much better on them in groups 1 & 4.

Deviance residuals in general were about the same or better than martingale residuals for the purpose of positive responder identification (i.e. negative nodes). The same was true for negative responders (positive nodes), except for data sets with large percentage of censoring. Overall, the responder identification algorithm using regression trees showed acceptable power of identification for data, in which the groups to be identified were with much larger (or much smaller) hazard than the entire data set (sim. groups 1, 4, & 7 with large predictive coefficients). The results were miserable for the data in sim. groups 3, 6, & 9, where the predictive coefficients were very small. This leads us to the conclusion, that regression trees are not sensitive enough method to be applied in responder identification. Nevertheless, if we had to make a recommendation which residuals to use as a response factor in CART, we would prefer deviance residuals, as they have acceptable performance at least for the case when censoring is not too large and the responder coefficients are strong (sim. groups 1, 2, 4, & 5). For data with large percentage censoring it is preferable to use martingale residuals.

9.2.3 Bump Hunting

Similar to section 9.2.2, the responder identification method from section 8.1 with original and stabilized bump hunting was applied on each of the 200 data sets of the nine simulation groups defined in table 9.1 (100 with martingale and 100 with deviance residuals as response in bump hunting).

Goal:

- 1) to evaluate the prognostic power of the responder identification algorithm (chapter 8) with bump hunting
- 2) to compare the power of identification of the method when martingale and deviance residuals are used as response variable in bump hunting, as well as when the original and the stabilized bump hunting procedure is used.

We began the responder identification process with the algorithm of section 8.1 as described in section 9.2.2. This time bump hunting (original and stabilized) was used instead of regression trees. It turned out, that in the simulated data the p-value of the log-rank statistic never grew insignificant in the stabilized and original bump hunting. We developed the bump hunting models using minimal support of .05 as stopping criteria (as suggested by Friedman & Fisher, 1999). Since we knew the correct responder and non-responder groups, we considered just one box per bump and just the first three selected borders. The developed model was considered correct if the first three borders of the maximal box were any permutation of the following:

$$X2 \neq 0, X2 \neq 1, X5 \neq 0$$

and the minimal box – any permutation of the following borders:

$$X4 \neq 0, X5 \neq 0, X6 \neq 0.$$

Table 9.5 summarizes an example of border selection with bootstrapping (100 bootstrap samples + original data) when the correct model was chosen. Figure 9.5 gives the support vs. mean of residuals and support vs. p(LR) for the maximal and the minimal box construction process of this particular example.

Table 9.5: *Example of border selection when the correct border was chosen and stabilized bump hunting was used as predictive model (100 bootstrap samples + original data).*

	X1≠0	X1≠1	X2≠0	X2≠1	X2≠2	X3≠0	X3≠1	X4≠0	X4≠1	X5≠0	X5≠1	X6≠0	X6≠1
max1	0	0	12	63	0	0	0	0	0	26	0	0	0
max2	0	0	101	0	0	0	0	0	0	0	0	0	0
max3	0	0	0	0	0	0	0	0	0	101	0	0	0
max4	0	0	0	0	0	0	0	0	32	0	0	0	69
max5	12	17	0	0	0	4	30	5	33	0	0	0	0
min1	0	0	0	0	0	0	0	43	0	10	0	48	0
min2	0	0	0	0	0	0	0	62	0	39	0	0	0
min3	0	0	0	0	23	0	0	0	0	78	0	0	0
min4	1	6	0	0	91	3	0	0	0	0	0	0	0
min5	7	12	46	3	0	31	2	0	0	0	0	0	0

Notice, that the correct three borders in the maximal box were the only ones chosen as first borders (max 1). Once border $X2 \neq 1$ is chosen, the other two correct borders were chosen in all bootstrap samples and the original data set (max 2 & max 3). For details on stabilized bump hunting, please refer to chapter 7. We know that the effect of the other two borders in the maximal box is by chance, since it was not programmed in the simulation. This also proves to be the case, as those borders change even in the stabilized procedure when a completely new data set is simulated (not shown in table 9.5). A weaker, but similar effect can be seen for the minimal box.

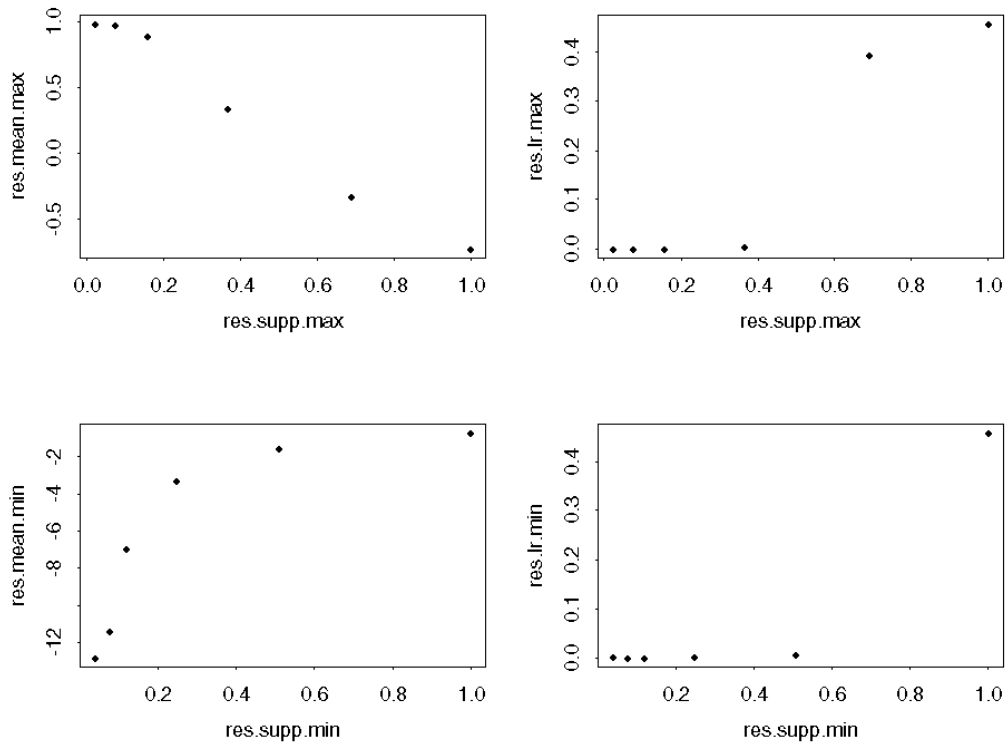


Figure 9.5: *Plots of mean response ($res.mean.min$ & $res.mean.max$) vs. support ($res.supp.min$ & $res.supp.max$) and p-value of the log-rank statistic ($res.lr.min$ & $res.lr.max$) vs. support for the box built in table 9.5.*

Naturally, the correct bumps are usually not known. One does not even know how many boxes each bump has. Fortunately, "real life" data sets are not as clean and ordered as our simulated data, so that the bump growth process can actually be governed by the

log-rank statistic, as described in chapters 7 & 8. A statistician's educated guess is still needed sometimes for altering the algorithm, as shown in chapter 10.5, where the stabilized bump hunting procedure was applied on the EMIAT data set in a responder identification process. An additional step lead to an improvement of the p-value of the log-rank statistic, which had grown in the previous step (see table 10.6).

Table 9.6 summarizes the results from all simulation runs. Each entry in the table represents the number of times the correct minimal or maximal bump was chosen from a total of 100 runs.

Table 9.6: *Number of times the correct minimal and maximal bump was chosen from 100 runs with original or stabilized bump hunting as predictive model and martingale or deviance residuals as response.*

sim. group #	MART				DEVI			
	MIN		MAX		MIN		MAX	
	original	stabilized	original	stabilized	original	stabilized	original	stabilized
1	71	97	95	99	42	72	100	100
2	65	88	98	100	33	52	100	100
3	47	84	94	99	26	46	100	100
4	72	99	98	99	39	78	100	100
5	67	92	97	99	35	59	99	100
6	52	79	97	100	22	36	100	100
7	75	97	96	99	43	78	100	100
8	67	89	94	99	31	48	100	100
9	56	83	94	98	26	35	99	100

The results seem to be independent of percent censoring: simulation groups (1, 4, 7), (2, 5, 8), and (3, 6, 9) have similar outcomes across the different methods. Size of the c_{\min} and c_{\max} coefficients show effect: larger in absolute value coefficients result in better performance of the different methods – please refer to table 9.7, which summarizes the results by coefficient size, averaged over all censoring cases. Martingale residuals show to be better suited for positive responder identification (minimal bump) than deviance residuals. For negative responder identification (maximal bump) deviance residuals perform just as well or slightly better than martingale residuals. In all cases where

improvement was possible, the stabilized bump hunting algorithm showed much better results than the original algorithm.

Table 9.7: *Averages from table 9.6 over simulation groups with equal percent censoring.*

sim. group #	MART				DEVI			
	MIN		MAX		MIN		MAX	
	original	stabilized	original	stabilized	original	stabilized	original	stabilized
1, 4, 7	72.67	97.67	96.33	99	41.33	76	100	100
2, 5, 8	66.33	89.67	96.67	99.33	33	53	99.67	100
3, 6, 9	51.67	82	95	99	24.67	39	99.67	100

Conclusions:

Deviance residuals perform excellent in negative responder identification and unsatisfactory in positive responder identification. Their use is not recommended when both responder groups are needed. The stabilized bump hunting procedure with martingale residuals as response variable delivers excellent results both in positive and negative responder identification, especially if the effect is strong.

9.3 Comparison

Table 9.8 gives a summary of the results of the responder identification algorithm when the best of regression trees and the best of bump hunting is employed (see tables 9.4 & 9.6).

Recall, comparison between a tree and a bump model can only be made in a very loose sense, since tree models describe the entire space and bump models – just extreme parts of it. In other words, the nature of bump models is much more adequate for responder identification.

Indeed, in 16 out of the 18 cases, bump hunting was more powerful than regression tree (as predictive model in the responder identification algorithm). In the two other cases the results of both models were comparable; regression tree performed slightly better than bump hunting. We can then conclude, that the best version of the responder

identification method employing regression trees is not nearly as good as the one with stabilized bump hunting. **It is, therefore, recommendable that the responder identification method, suggested in chapter 8, be used with martingale residuals as a response variable in the stabilized bump hunting (described in chapter 7).**

Table 9.8: *Number of correctly identified models from 100 runs in the nine simulation groups using the best of the responder identification algorithms employing regression trees and bump hunting.*

sim. group #	MIN		MAX	
	TREE + DEVI	stable BUMP + MART	TREE + DEVI	stable BUMP + MART
1	100	97	65	99
2	94	88	71	100
3	52	84	10	99
4	94	99	81	99
5	76	92	70	99
6	33	79	6	100
7	59	97	52	99
8	53	89	32	99
9	10	83	1	98

9.4 Implementation

This simulation study was performed with the help of the readily available statistical packages SPSS and S-PLUS and the programming languages S and C. The Cox model with interactions and the Kaplan-Meier curves were generated using the survival analysis tools in SPSS 10.0. The simulation of all data sets, as well as the bump hunting analysis were done with especially written for the purpose S programs, which run with S-PLUS 4.5. Construction of the bump models was done in S-PLUS for Unix, using the algorithms of Becker (1999) called `.boxes`, `.express.boxes`, and `.border.ranking`, which use C subroutines. The part of the simulation study involving regression tree models was done in S-PLUS for Windows. S routines using the S-PLUS tools for regression tree construction were written for that purpose. The code of all self-implemented S routines are given in Appendix C.

10. APPLICATIONS: EMIAT

10.1 Data

The European Myocardial Infarction Amiodarone Trial (EMIAT) is a randomized double blind placebo controlled trial, designed to compare the drug Amiodarone to placebo with respect to all cause mortality. It includes a total of 1486 survivors of acute myocardial infarction who have left ventricular ejection fraction (LVEF) of 40% or less, randomized into two groups of 743 patients each. There were 103 deaths in the Amiodarone arm and 102 deaths in the placebo arm of the study. A total of 1169 patients had Holter recordings available with sinus rhythm and at least one ventricular premature beat (VPB), which are necessary for calculation of the new parameters Onset and Slope – the two components of heart rate turbulence (HRT)¹. The Amiodarone group had 577 patients, 87 of which died during the two years of follow-up (85% censoring). 592 patients were in the placebo group, 82 of which died (86% censoring). Figure 2.1 shows the Kaplan-Meier survival function estimates in the two study arms. Visually, as well as statistically, no difference between the two curves can be found ($p(\text{LR}) = .5815$). Baseline patient characteristics can be found in table 10.1. Continuous factors were categorized as shown in the table, using cut points, chosen by the EMIAT investigators and the research group of Prof. G. Schmidt at the Technical University in Munich.

10.2 Previous investigations

Janse et al (1998) did subgroup analysis of the EMIAT data in order to find patients, who may benefit from treatment with Amiodarone, i.e. they were looking for positive responders. The strategy performed in this substudy of EMIAT was to choose four

¹ Updated information on parameter HRT can be found at www.h-r-t.com

important, readily available baseline characteristics and consider all groups resulting from their combinations. The factors chosen were:

Left ventricular ejection fraction, dichotomized at 30%
 Arrhythmia signs on Holter recordings (Yes/No)
 Beta-blocker treatment (Yes/No)
 Heart rate (Low/High)*

In both arms of the study:

Amiodarone treatment (Yes/No (placebo))

* *The lowest and the highest 25% of the heart rate measurements were used; cut points were determined separately for each subgroup, defined by a combination of the rest of the factors. A total of 80 subgroups were analyzed.*

All possible subgroups were considered. In each group, the event rates in the placebo and the Amiodarone arms were compared. The log-rank statistic was computed.

The largest reduction of event rate on Amiodarone vs. placebo (i.e. positive responders) was found for the group:

ARRHYTHM = Yes
 Beta-blocker = Yes
 Heart rate = High (≥ 75 beats/min)

$p(\text{LR}) = 0.15$ (not significant at the 0.05 level)

The largest increase of event rate on Amiodarone vs. placebo (negative responders) was found for the group:

LVEF $\geq 30\%$
 Beta-blocker = No
 Heart rate = Low (≤ 66 beats/min)

$p(\text{LR}) = 0.0314$ (significant at the 0.05 level)

Notice, that only interactions of up to third order were considered. No adjustment for prognostic factors was done. For details, see Janse et al, 1998.

Table 10.1: *Baseline characteristics of EMIAT.*

Variable	Code Name	Dichotomization	Placebo (n = 592)		Treatment (n = 577)	
			Mean (SD)	Number (%)	Mean (SD)	Number (%)
Sex	SEX	1 = male		506 (86%)		487 (84%)
More than one infarct	INFARCT	1 = Yes		157 (27%)		188 (33%)
New York Heart Association Classification	NYHA	1 2 3		251 (42%) 44 (7%)		264 (46%) 46 (8%)
Diabetes	DIABETES	1 = Yes		95 (16%)		98 (17%)
Thrombolyse	THROMBOL	1 = Yes		355 (60%)		321 (56%)
Digoxin	DIGOXIN	1 = Yes		73 (12%)		86 (15%)
β - blocker	BETABLO	1 = Yes		262 (44%)		255 (44%)
Calcium-antagonist	CALCANT	1 = Yes		81 (14%)		71 (12%)
ACE - inhibitors	ACEINHI	1 = Yes		348 (59%)		354 (61%)
Arrhythmia on Holter	ARRHYTHM	1 = Yes		208 (35%)		212 (38%)
Left-ventricular ejection fraction	LVEF	1 if LVEF \leq 30	29.92 (7.52)	278 (47%)	30.20 (6.99)	275 (48%)
Age	AGE	1 if AGE $>$ 65	60.62 (9.33)	240 (41%)	60.21 (9.67)	220 (38%)
Mean heart rate frequency	FREQ	1 if FREQ $>$ 75	73.37 (11.76)	250 (42%)	73.10 (12.02)	251 (44%)
Heart rate variability index	HRVI	1 if HRVI \leq 20	26.08 (10.38)	185 (31%)	26.34 (10.41)	175 (30%)
Onset	ONSET	1 if ONSET $>$ 1	0.99 (0.023)	158 (27%)	0.99 (0.026)	148 (26%)
Slope	SLOPE	1 if SLOPE \leq 2.5	6.60 (8.08)	172 (29%)	6.43 (8.39)	189 (33%)
Heart Rate Turbulence	HRT	0 if ONSET = 0 & SLOPE = 0		174 (29%)		181 (31%)
		1 if ONSET = 1 or SLOPE = 1 2 if ONSET = 1 & SLOPE = 1		78 (13%)		78 (14%)

Malik et al (2000) performed subgroup analysis of the EMIAT data set with final aim to test the hypothesis that EMIAT patients with depressed heart rate variability (HRV) benefit from the Amiodarone treatment (i.e. are positive responders). They did this by developing a Cox-PH model on the entire data set, including Amiodarone treatment as a factor, which accounts for the following prognostic factors:

Age, dichotomized at 60 years

LVEF, dichotomized at 30%

History of MI (Yes/No)

Heart rate, dichotomized at 75 beats/min

Arrhythmia on Holter (Yes/No)

Beta-blocker treatment (Yes/No)

HRV index, dichotomized at 20 units

The authors applied this model on various subgroups of the data and concluded, that in groups of patients with depressed HRV, the Amiodarone treatment factor increases its significance. Table 10.2 gives a very brief summary of their findings. For further details, please refer to the original publication.

Table 10.2: *Summary of the results of Malik et al (2000).*

Group	p-value of Amiodarone treatment after accounting for prognostic factors
Total population	.9068
Depressed HRV	.3221
Depressed HRV & Low LVEF	.3785
Depressed HRV & No history of MI	.0417*
Depressed HRV & High heart rate	.2404
Depressed HRV & Arrhythmia	.1458
Depressed HRV & on Beta-blocker	.4909

** significant at the 0.05 level*

10.3 Cox-PH with interaction

The following results were obtained after applying the method described in chapter 4 on the EMIAT data set. Continuous factors were not dichotomized.

Using forward selection on the entire data set, the model summarized in table 10.3 was found as "best."

The search for interactions delivered just one possible predictive factor, heart rate (HR_V0), which was not significant at the .05 level but, nevertheless, increased the significance of factor treatment from .78 (in a model including prognostic factors only) to .08. The interaction model is summarized in table 10.4.

Table 10.3: *Summary of the "best" Cox-PH model without interactions on the entire EMIAT data set.*

Variables	β	p(Wald)	Exp(β)	95% CI for Exp(β)	
				Lower	Upper
LVEF	-.032	.002	.968	.949	.988
AGE	.030	.003	1.031	1.010	1.051
INFARCT	.737	.000	2.090	1.534	2.849
DIABETES	-.454	.009	.635	.452	.893
HR_V0	.016	.003	1.017	1.006	1.028
HRT		.001			
HRT(1)	.439	.022	1.551	1.064	2.261
HRT(2)	.842	.000	2.320	1.506	3.575

score statistic = 148.49
p(score) < .001

Table 10.4: *Summary of the "best" Cox-PH model with interactions on the entire EMIAT data set.*

Variables	β	p(Wald)	Exp(β)	95% CI for Exp(β)	
				Lower	Upper
LVEF	-.032	.001	.967	.948	.987
AGE	.031	.002	1.031	1.011	1.052
INFARCT	.734	.000	2.084	1.528	2.841
DIABETES	-.464	.008	.629	.447	.884
HR_V0	.025	.001	1.025	1.011	1.040
HRT		.001			
HRT(1)	.434	.024	1.544	1.059	2.252
HRT(2)	.842	.000	2.321	1.506	3.578
TREATMEN	1.420	.083	4.137	.830	20.626
HR_V0*TREATMEN	-.017	.087	.983	.964	1.002

score statistic = 150.60
p(score) < .001

Analyzing the part of the linear predictor corresponding to the interaction term in the last model, i.e. the linear predictor for factors HR_V0, TREATMEN, and their interaction, it is easy to see that, in general, increase in HR_V0 results in increase of the hazard. A treatment with Amiodarone decreases this effect (please refer to figure 10.1). The lines cross at about HR_V0 = 83, so that patients with heart rate greater than 83 should be positive responders and ones with heart rate less than 83 – negative responders. However, looking at the survival curves of the two groups in both treatment arms, we find the difference to be not significant at the .05 level ($p(\text{LR}) = .3058$ for $\text{HR_V0} \leq 83$ and $p(\text{LR}) = .6045$ for $\text{HR_V0} > 83$).

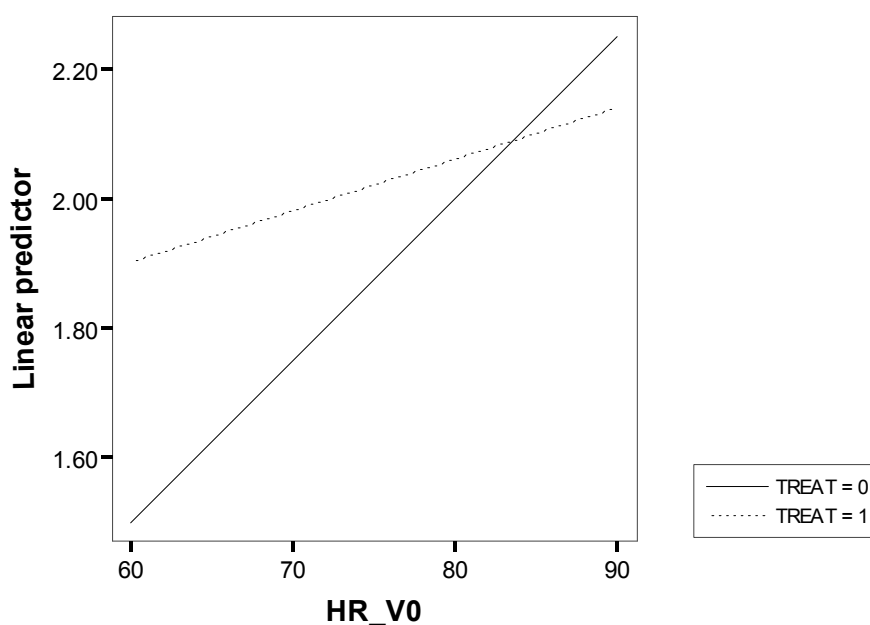


Figure 10.1: *Interaction of factors HR_V0 and treatment ($TREAT = 0$ denotes placebo, $TREAT = 1$ denotes Amiodarone).*

On the other hand, figure 2.2 in chapter 2.1 shows different effects of treatment with beta-blocker on mortality in the two arms of EMIAT. One should expect to see interaction between beta-blocker and Amiodarone treatment, but it does not appear in the predictive part of the Cox model. Considering the fact that the EMIAT data set has even higher percentage censoring than data from group 9 of the simulation study (chapter 9), and knowing how badly the Cox-PH model with treatment interaction performed on such data (see section 9.2.2), one should not expect great results on the EMIAT data. Cox-PH model with interactions is simply not a sufficiently good way of doing responder analysis, even though it is more systematic than the work of Janse et al and Malik et al.

10.4 Responder identification with CART

10.4.1 The prognostic model

As described in chapter 8, one needs to develop a "good" prognostic model in the first step of the responder identification algorithm. We developed a Cox-PH model on the placebo group of EMIAT, using stepwise selection methods and validated it internally. The "good" prognostic model we found contains the continuous factors left-ventricular ejection fraction (LVEF) and heart rate at initial visit (HR_V0), and the categorical factors previous infarction (INFARCT) and Heart Rate Turbulence (HRT). The model was internally validated using 100 bootstrap samples of the placebo group. When applied to the samples, the Cox model had mean score statistic of 78.68 (SE = 20.64), which is in the same order as the one from the original sample, even slightly better. A summary of the chosen prognostic model is given in table 10.5.

Table 10.5: *Summary of the "best" Cox model on the placebo group of EMIAT.*

Variable	β	p(Wald)	Exp(B)	95.0% CI for Exp(B)	
				Lower	Upper
LVEF	-0.035	.013	0.966	0.939	0.993
HR_V0	0.024	.001	1.024	1.009	1.039
INFARCT	0.603	.008	1.827	1.171	2.850
HRT		.000			
HRT(1)	0.599	.030	1.820	1.059	3.128
HRT(2)	1.181	.000	3.257	1.818	5.835

score statistic = 69.83
p(score) = 1.14 × 10⁻¹³

Further, the hazard function and martingale residuals in the Amiodarone group were calculated using the baseline hazard function and the factor coefficients as estimated in the placebo group Cox model.

10.4.2 The predictive model with continuous factors

The use of continuous vs. categorized factors in regression trees was discussed in section 8.4. The simulation study in chapter 9 considered the only continuous factor (X3) dichotomized at the mean. It is well known, that categorization of continuous factors limits the recursive partitioning process and decreases the goodness of fit of the resulting model, however, it also increases the predictive power of the model, which is often the more desirable quality of both. In the following two sections, we will build regression tree models with both types of factors, which we will then compare.

Regression tree analysis was performed on the Amiodarone arm of EMIAT, where all available factors (see table 10.1) were used as predictors. Since EMIAT has very large percent censored cases, we used martingale residuals as response of the tree model, as recommended in chapter 9 on the basis of a simulation study. Continuous factors were not categorized here. Initially, a large tree was grown, after which it was pruned down to a tree with ten end nodes, using pruning parameter $\alpha = 1.85$ (see chapter 5 for details on pruning). Figure 10.2 gives the tree diagram of the final tree – our predictive model, containing factors SLOPE, FREQ, NYHA, AGE, and DIGOXIN.

In order to find responders, we consider the end nodes one by one, ordering them by the size of their mean response (martingale residuals), as described in section 8.3. We split the placebo group into ten regions, defined as the end nodes of the regression tree, which was constructed on the Amiodarone group. Now we are able to describe each region in the Amiodarone arm and compare it to its corresponding region in the placebo arm.

Consider first all "negative" nodes, i.e. nodes with negative mean of the residuals (figure 10.2). We will be looking for responders in them. Starting with the end node which has the largest negative mean of the residuals in the Amiodarone group (node 14, region R6), we plot the mean of the residuals and its 95% confidence interval (figure 10.3). There are only eight patients in R6 (Amiodarone arm) and the log-rank statistic comparing patients in R6 in the Amiodarone and the placebo groups is not significant at the .05 level ($p(\text{LR}) = .1187$). Further, we can combine the two end nodes with the

largest negative mean of the residuals in the Amiodarone group (node 14, region R6 and node 19, region R10). The mean and 95% confidence interval of the combined regions in the treatment group is also plotted in figure 10.3. The process is repeated until no more negative end nodes are left (four steps). The results are summarized in figure 10.3. Since initially there was no difference in survival between the Amiodarone and placebo groups (the entire groups), in our search for responders we should stop at the combination of end nodes in figure 10.3, just before the log-rank statistic becomes insignificant or when its p-value stops decreasing, as discussed in section 8.3. In our case, this happens for the combination of regions R6, R10, and R4, $p(\text{LR}) = .0036$.

Then patients having the characteristics of one of these regions would be considered to be **positive responders**:

R4:	SLOPE < 1.5697 FREQ ≥ 83.5 AGE < 56	R6:	SLOPE < .66004 FREQ ≥ 83.5 62.5 ≤ AGE < 70	R10:	.66004 ≤ SLOPE < 1.5697 FREQ ≥ 83.5 AGE ≥ 70.5 DIGOXIN = 2 (No)
------------	---	------------	--	-------------	--

In order to find negative responders, the process described above needs to be repeated for all "positive" end nodes, starting with region R8, which has the highest mean of the residuals in the Amiodarone arm. Figure 10.4 illustrates the six step process.

We can conclude that regions R2, R5, R7, R8, and R9 define **negative responders**:

R2:	SLOPE < 1.5697 FREQ < 83.5 NYHA = 1	R5:	SLOPE < 1.5697 FREQ ≥ 83.5 56 ≤ AGE < 62.5	R7:	SLOPE < .66004 FREQ ≥ 83.5 AGE ≥ 70
R8:	SLOPE < 1.5697 FREQ ≥ 83.5 AGE ≥ 62.5 DIGOXIN = 1 (Yes)	R9:	SLOPE < .66004 FREQ ≥ 83.5 62.5 ≤ AGE < 70.5 DIGOXIN = 2 (No)		

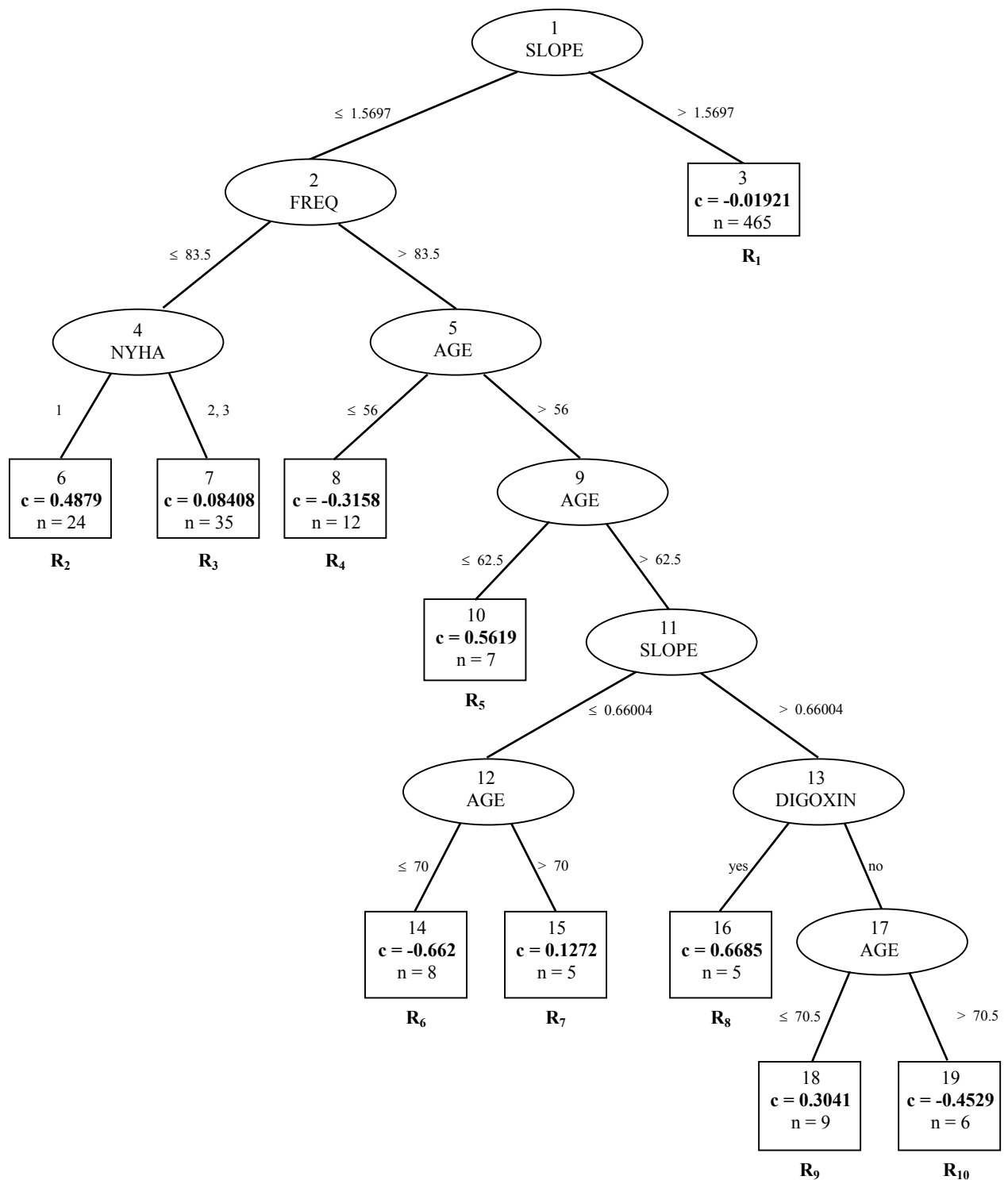


Figure 10.2: *Predictive regression tree model on the Amiodarone arm of EMIAT with continuous input factors and martingale residuals of the prognostic model in section 10.4.1 as response variable.*

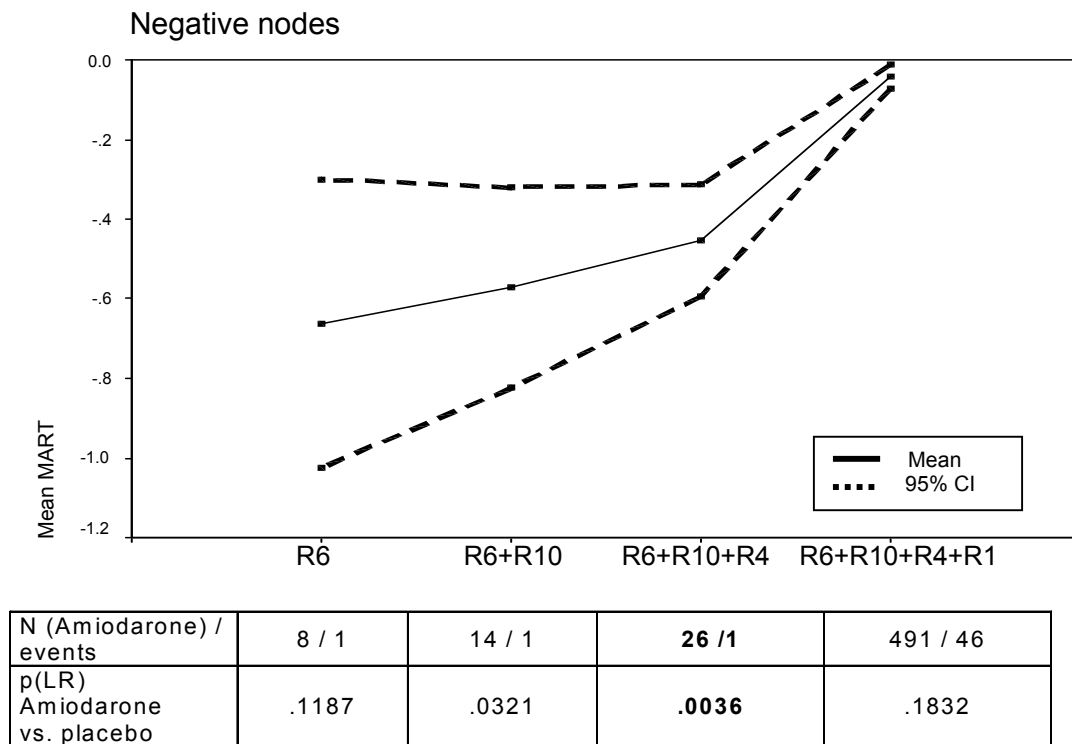


Figure 10.3: *Growth of the positive responder group (continuous factors tree).*

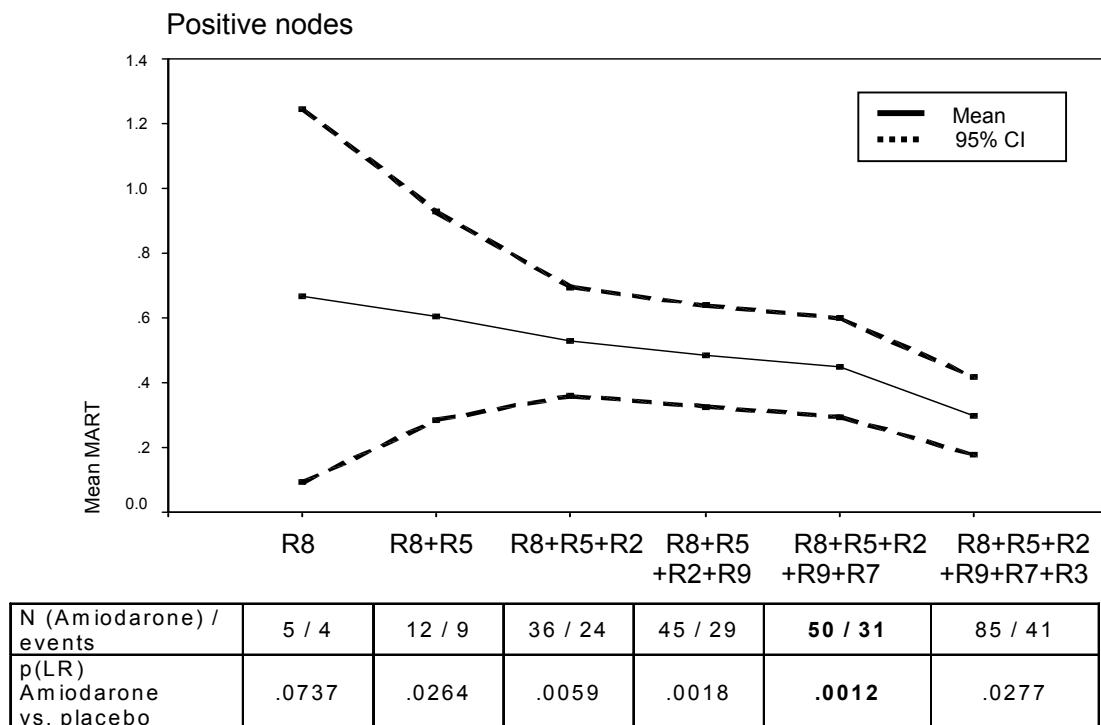


Figure 10.4: *Growth of the negative responder group (continuous factors tree).*

The scatter plot of figure 10.5 shows the martingale residuals of the identified positive and negative responders in the Amiodarone and placebo arm of EMIAT.

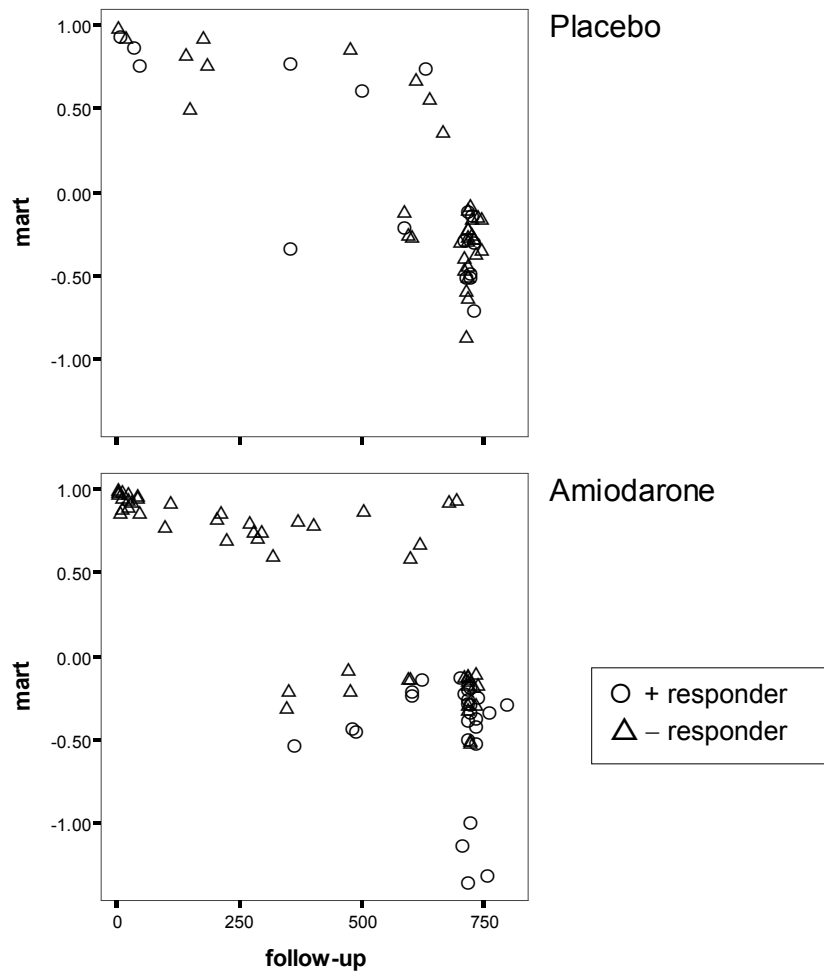


Figure 10.5: Scatter plot of the residuals of all patients in the responder groups for the placebo and Amiodarone arms of EMIAT (continuous factors tree).

$R6+R10+R4$ = positive (+) responders

$R8+R5+R2+R9+R7$ = negative (-) responders

10.4.3 The predictive model with categorized factors

The size of the EMIAT data set and most of all its high percent censoring do not allow for internal validation, so in order to develop a more stable predictive model with CART, one can use pre-defined cut points. Pros and cons of factor categorization are discussed in sections 7.1 and 8.4. The resulting regression tree model with predefined cut points (as in table 10.1) was pruned down to a tree with 13 end nodes, using pruning parameter $\alpha = .925$ (see chapter 5 for details on pruning). Figure 10.6 gives the tree diagram of the pruned tree, which contains the following predictive factors: ONSET, AGE, DIABETES, HRVI, BETABLO, LVEF, SLOPE, NYHA, FREQ, SEX and ARRYTHM. The search for end nodes containing responders is repeated as in section 10.4.2.

We consider first all end nodes containing Amiodarone patients with negative mean martingale residuals. Starting with node 19, region R6, which has the largest negative mean of the residuals (figure 10.6), we plot the mean and its 95% confidence interval in figure 10.7. Next, we plot the combined mean of the two end nodes with largest mean of the residuals (calculated in the Amiodarone group), namely node 19, region R6 and node 10, region R3. We repeat this process a total of six times until no more negative end nodes are available. The survival curves of the patients in each of the given end node combinations are compared between the placebo and Amiodarone groups. The p-values of the resulting log-rank statistics are also given in figure 10.7. Judging the end node combinations by their size, mean of the residuals, and p-value of the log-rank statistic, we would choose the group of regions R6, R3, R10, and R13 as the positive responder group. That means, that the regression tree predictive model of figure 10.5 defines patients with the following characteristics to be **positive responders** of Amiodarone (one factor combination should hold):

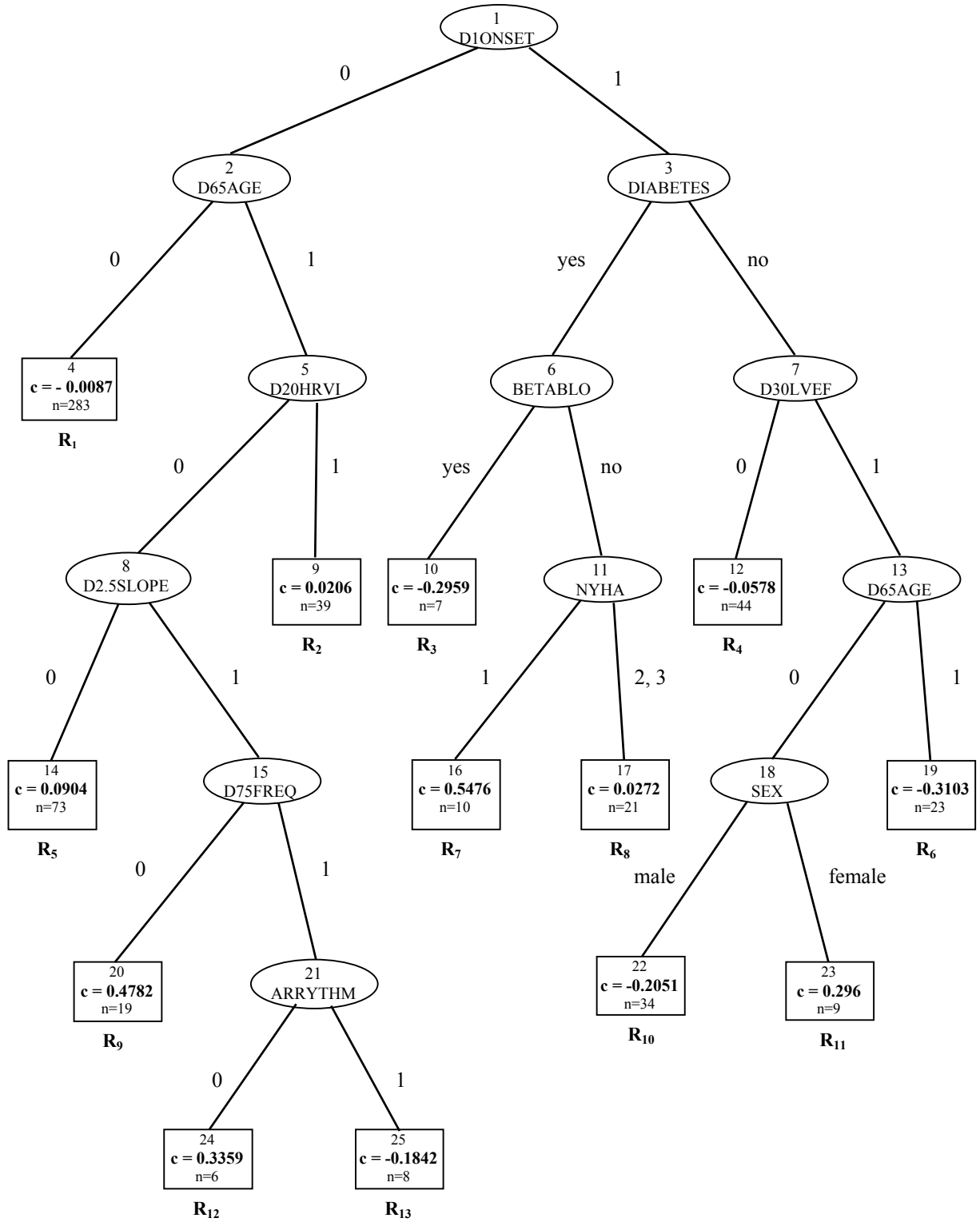


Figure 10.6: *Predictive regression tree model on the Amiodarone arm of EMIAT with categorized input factors and martingale residuals of the prognostic model in 10.4.1 as response variable.*

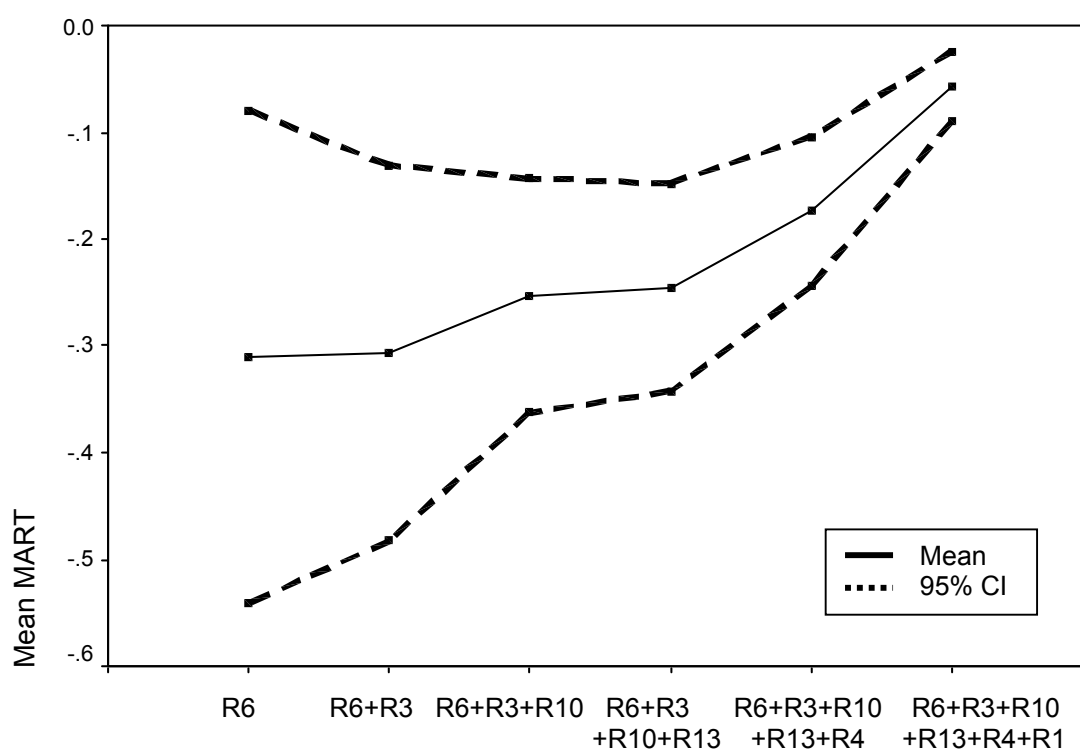
R6: ONSET > 1
DIABETES = No
LVEF \leq 30
AGE > 65

R3: ONSET > 1
DIABETES = Yes
BETABLO = Yes

R10: ONSET > 1
DIABETES = No
LVEF \leq 30
AGE \leq 65
SEX = Male

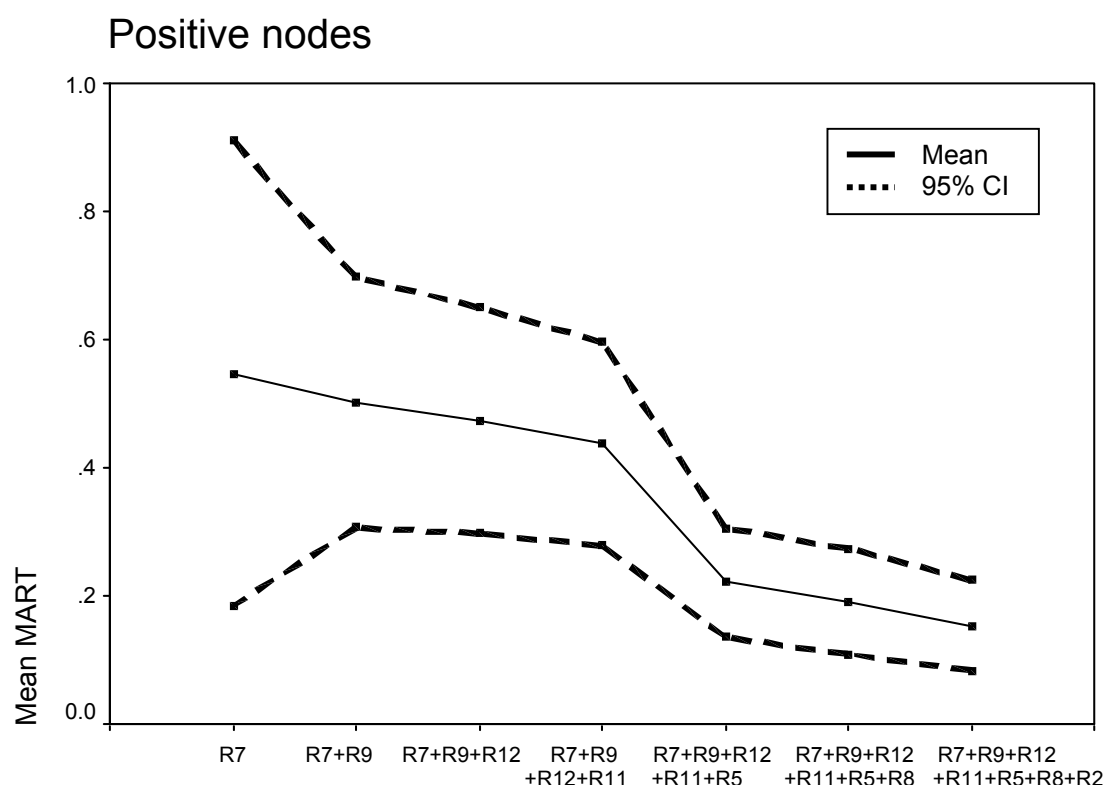
R9: ONSET \leq 1
AGE > 65
HRVI > 20
SLOPE \leq 2.5
FREQ > 75
ARRYTHM = Yes

Negative nodes



N (Amiodarone) / event	23 / 3	30 / 3	64 / 6	72 / 6	116 / 10	399 / 34
p(LR) Amiodarone vs. placebo	.1394	.0449	.0058	.0015	.0053	.1239

Figure 10.7: Growth of the positive responder group (categorized factors tree).



N (Amiodarone) / event	10 / 8	29 / 19	35 / 22	44 / 26	117 / 39	138 / 46	177 / 53
p(LR) Amiodarone vs. placebo	.1026	.0082	.0171	.0058	.0047	.0126	.0186

Figure 10.8: *Growth of the negative responder group (categorized factors tree).*

The same procedure is repeated on the positive end nodes in the search for non-responders. The results are shown in figure 10.8. The p-values of the log-rank statistic are of the same order for all shown end node combinations except the first one (R7). We choose the last region combination before the mean of the martingale residuals dramatically drops, namely, regions R7, R9, R12, and R11. The difference in survival between patients with the following characteristics in the Amiodarone and the placebo arm is significant at the .05 level ($p(\text{LR}) = .0058$):

R7:	ONSET > 1 DIABETES = Yes BETABLO = No NYHA = 1	R9:	ONSET ≤ 1 AGE > 65 HRVI > 20 SLOPE ≤ 2.5 FREQ ≤ 75
------------	---	------------	--

R12: ONSET ≤ 1
 AGE > 65
 HRVI > 20
 SLOPE ≤ 2.5
 FREQ > 75
 ARRYTHM = No

R11: ONSET > 1
 DIABETES = No
 LVEF ≤ 30
 AGE ≤ 65
 SEX = Female

If a patient is in one of the groups above, he/she would be considered a **negative responder** to Amiodarone. The residuals of the positive and negative responders are plotted against follow-up time in figure 10.9.

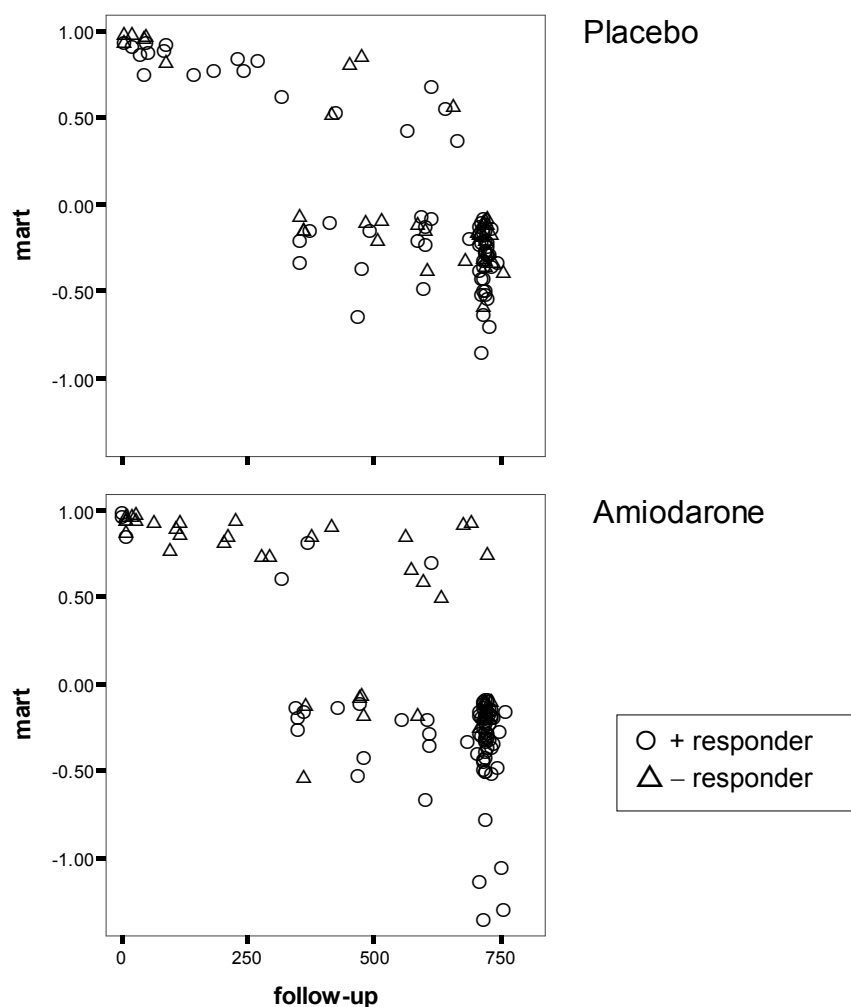


Figure 10.9: Scatter plot of the residuals of all patients in the responder groups for the placebo and Amiodarone arms of EMIAT (categorized factors tree).
 $R6+R3+R10+R13$ = positive (+) responders
 $R7+R9+R12+R11$ = negative (-) responders

Notice, that the p-value of the log-rank statistic for the two largest positive end nodes is .0082. Adding one more node increases the p-value. Instead of stopping here, as suggested by the algorithm, we made one more step, which did not change the mean of the residuals in the group, but decreased the p-value dramatically. Sometimes it is worth to look a step ahead in the algorithm.

As it should be expected, the regression tree with continuous covariates finds groups of Amiodarone patients with more difference in survival (when compared to similar patients under placebo) than the model with categorized factors. For fair comparison, consider equal size groups:

26 negative responders identified with **continuous** factors bring $p(\text{LR}) = .0036$.

26 negative responders identified with **categorical** factors bring
 $.0449 < p(\text{LR}) < .1394$ (for $N = 30$ & 23 respectively).

50 positive responders identified with **continuous** factors bring $p(\text{LR}) = .0012$.

50 positive responders identified with **categorical** factors bring
 $.0047 < p(\text{LR}) < .0058$ (for $N = 115$ & 44 respectively).

Table 10.6: *Identified responders errors in the algorithm with continuous and categorized factors regression trees. Table cells represent number of patients in the Amiodarone arm.*

		CONTINUOUS TREE			Total
		+ responders (-1)	non-responders (0)	- responders (1)	
CATEGORIZED TREE	+ responders (-1)	9	52	11	72
	non-responders (0)	16	420	25	461
	- responders (1)	1	29	14	44
Total		26	501	50	577

Table 10.6 gives a comparison in responder identification of the tree models with continuous and categorized factors in the Amiodarone group. The patients in the main nine cells of table 10.6 are also represented with their martingale residuals in figure 10.10. The crucial mismatches in classification with respect to the other model are depicted in the plots of (row 1, column 3) and (row 3, column 1). The first shows the 11

Amiodarone patients who were considered to be positive responders by the categorized factors model and negative responders – by the continuous factors model. The second shows the patient who was considered to be a positive responder by the continuous and negative responder – by the categorized factors model. In both cases the continuous factors model shows, as expected, better fit to the data. However, the cut points in the continuous factors tree are hierarchically dependent on all higher level nodes, which makes them impossible to reproduce when the model is built on a slightly altered or new data set. Therefore, it is preferable to use the categorized factors tree model for responder identification purposes.

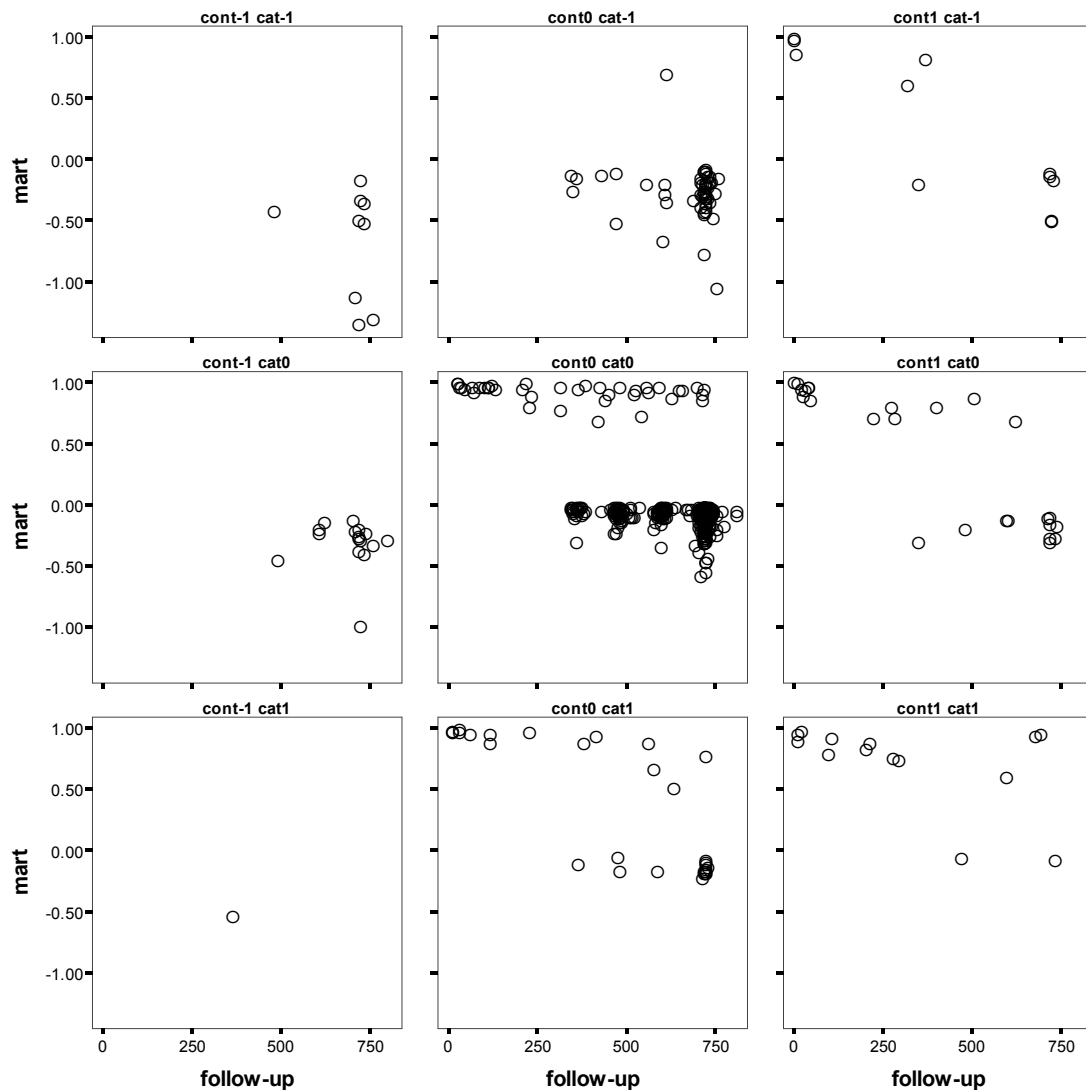


Figure 10.10: *Martingale residuals of the Amiodarone patients in groups as in table 10.6. 1 = – responders, 0 = non-responders, -1 = + responders.*

10.5 Responder identification with bump hunting

As shown in the simulation study of chapter 9, the stabilized through bootstrapping bump hunting procedure with response martingale residuals delivers the best results in the responder identification algorithm. The following is an application of that algorithm on the EMIAT data set.

We use the prognostic model developed in section 10.4.1 on the placebo arm of EMIAT with its martingale residuals in the Amiodarone as a response variable in the following stabilized bump hunting application. Before the search for predictive bump model can begin, all continuous variables need categorization (as in table 10.1).

When the stabilized bump hunting procedure is applied on the Amiodarone arm of EMIAT, it finds the following predictive "bump" model:

MAX bump		MIN bump
<u>Box 1</u>	<u>Box 2</u>	<u>Box 1</u>
	<i>All not in Box 1 ∩:</i>	<i>All not in MAX bump ∩:</i>
ONSET ≤ 1	DIABETES = 1	CALCANT = 0
AGE ≥ 65	BETABLOC = 0	DIABETES = 0
NYHA > 1	THROMBOL = 0	ONSET > 1
	CALCANT = 0	SEX = male

The maximal bump contains two boxes – one with three borders and one with four borders. The minimal bump consists of a single box with four borders. The evolution of the support-mean relationship of the bump model is shown in figure 10.11. Notice, that the support-mean points are denoted with circles for the three borders of the first positive box. Then Box 2 is added to the model border by border (denoted with triangles). Finally, the negative bump is added to the model (the four rhombs). The final support-mean of the model is denoted by the triangle with smallest support and largest mean for the positive bump and the rhomb with smallest support and largest in absolute value mean for the negative bump. The growth of the bump model is shown in detail in table 10.7, including the p-value of the log-rank statistic when comparing patients with the current model characteristics in the placebo vs. Amiodarone groups. Notice, that in the minimal box, adding the third border actually slightly increases the p-value of the

log-rank statistic, however, adding the fourth border decreases it dramatically. As described in chapter 7, the bootstrapped bump hunting procedure stops when the p-value does not improve from the previous step. In our case, it was worth to consider one additional step. In this case, we used the advantages of doing responder analysis with a semi-automated software implementation to overcome the nearsightedness of the p-value stopping criteria.

Both the maximal and minimal bumps define groups of patients who have significantly different survival estimates under Amiodarone and under placebo. Therefore, we can consider the positive bump as a definition of negative responders and the negative bump as a definition of positive responders of Amiodarone under the conditions of the EMIAT study. Figure 10.12 contains a scatter plot of martingale residuals vs. follow-up time of the identified positive and negative responder groups, which shows the possibly misclassified by the model patients, i.e. positive responders with positive residuals and negative responders with large negative residuals in the Amiodarone panel. Those would be patients, for whom the prognostic and the predictive factors in the chosen models do not explain the changes in survival pattern under Amiodarone. As discussed in chapter 2, they either appeared by chance, or some prognostic and/or predictive factors were not accounted for in the EMIAT study and, therefore, the effect of Amiodarone on those patients cannot be explained by the current models. The "flower" plot of figure 10.13 represents schematically the structure of the positive and negative responder groups. Figure 10.14 shows the Kaplan-Meier survival curve estimates for those groups in the Amiodarone and placebo arms of EMIAT.

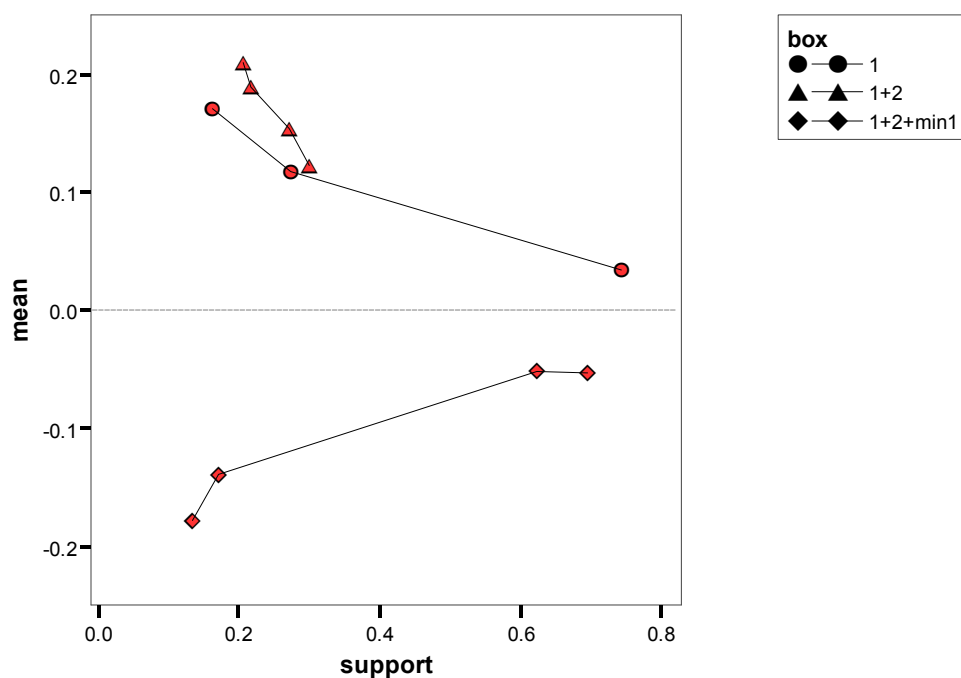


Figure 10.11: *Growth of the stabilized bump hunting model in the Amiodarone arm of EMIAT.*

Table 10.7: *Cumulative bump means and p-values stepwise after the addition of each new border to the bump model (to complement figure 10.11).*

			PLACEBO			AMIODARONE		
		p(LR)	Mean	n	events	Mean	n	events
MAX	1.	.1537	.0298	434	44	.0341	428	57
Box 1	2.	.1622	.0543	164	26	.1170	158	36
	3.	.1007	.0561	86	15	.1708	93	27
MAX	1.	.1181	.0457	165	33	.1220	172	48
Box 2	2.	.0607	.0474	146	30	.1538	156	48
	3.	.0334	.0507	117	24	.1887	125	42
	4.	.0147	.0453	115	23	.2096	119	42
MIN	1.	.0614	-.0197	413	53	-.0529	400	35
Box 1	2.	.0979	-.0067	372	45	-.0523	359	30
	3.	.1037	-.0142	105	22	-.1398	98	12
	4.	.0236	-.0008	89	20	-.1788	77	7

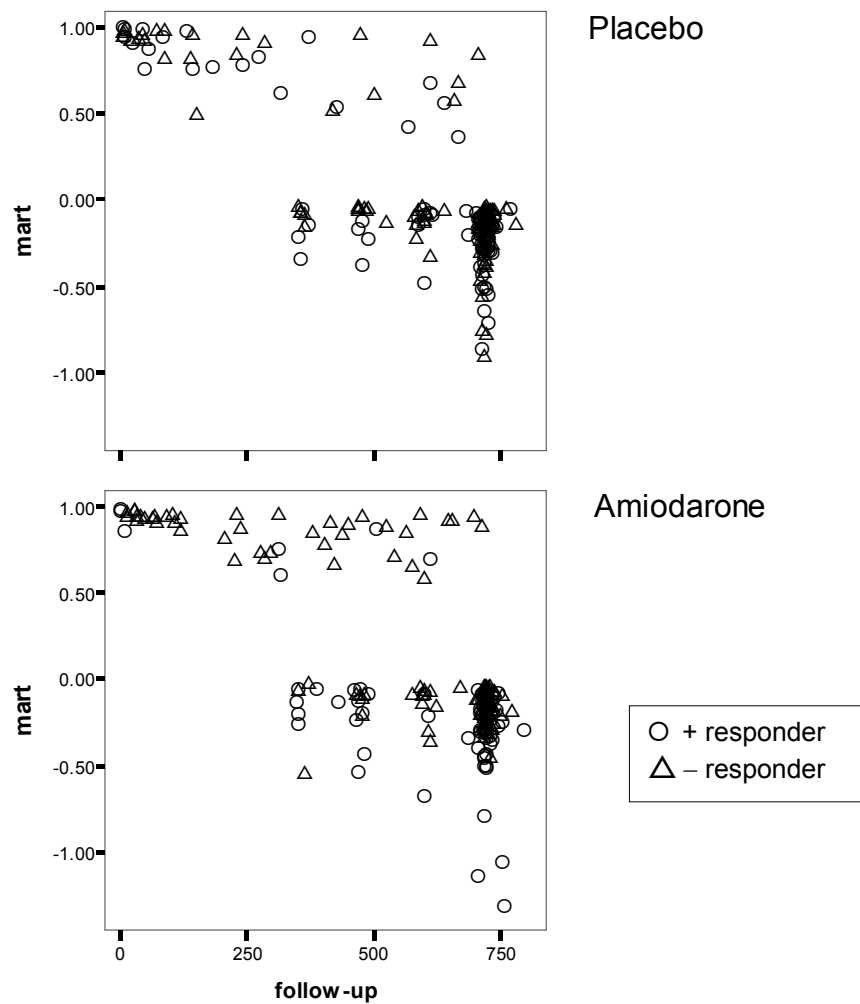


Figure 10.12: *Scatter plot of the residuals of all patients in the responder groups for the placebo and Amiodarone arms of EMIAT (stabilized bump hunting model).*

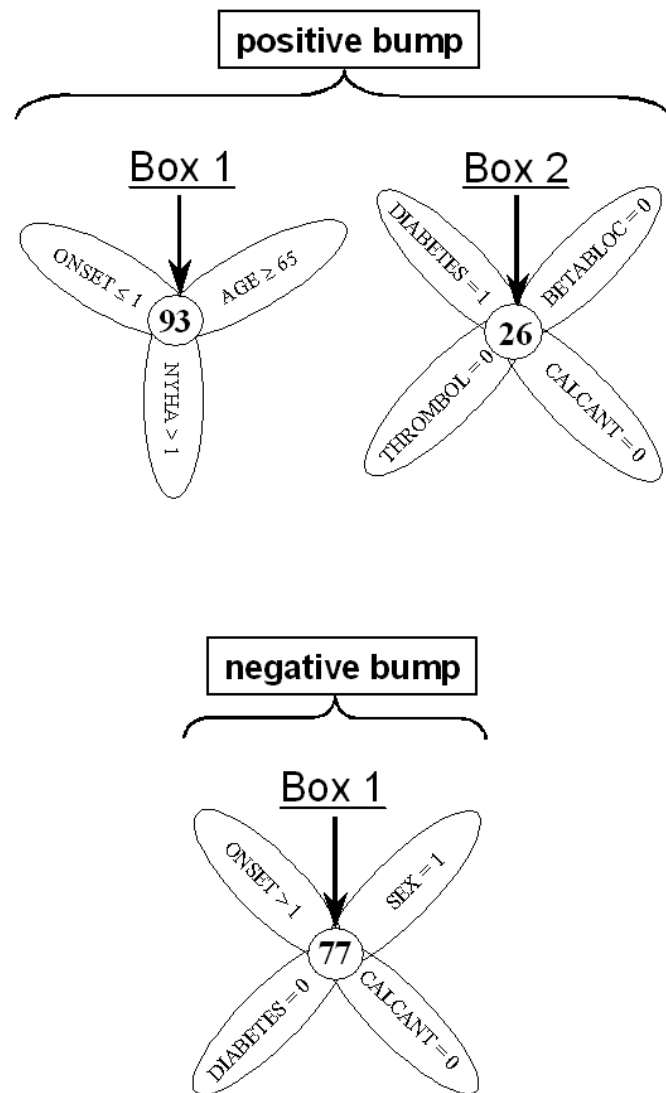


Figure 10.13: *Flower plot of the bump model.*

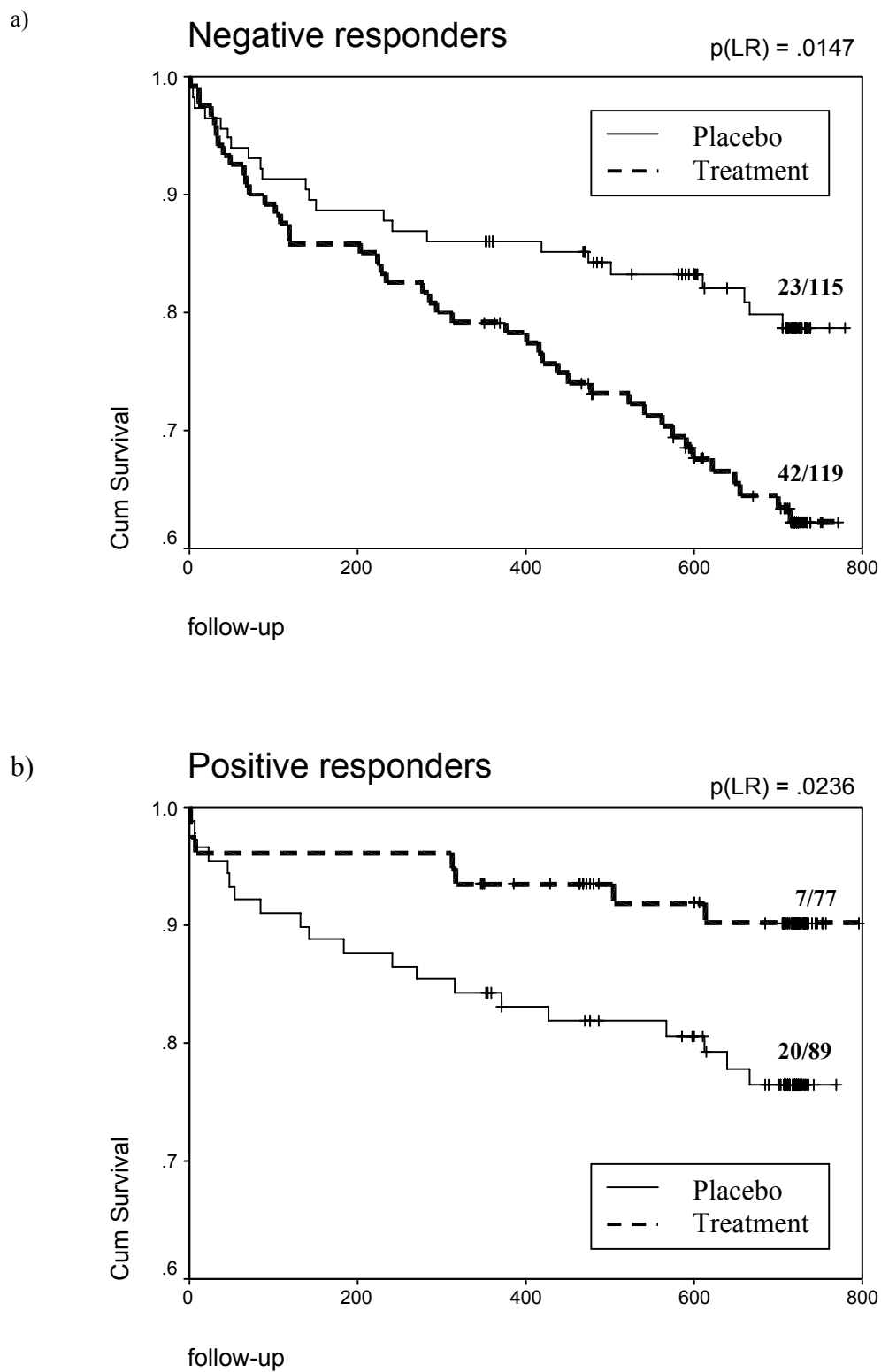


Figure 10.14: *Kaplan-Meier survival curve estimates for the two responder groups of the bump model, compared in the Amiodarone and placebo groups of EMIAT.*

10.6 Comparison and discussion

From chapter 9 we know, that the bump hunting model performs better than the tree model in responder identification context when all factors are categorized. Nevertheless, let us compare the outcomes of the categorical factor regression tree and the stabilized bump hunting models. We know from section 10.4.3, that the continuous factors tree model fits better the data. We also know from chapter 9, that stabilized bump hunting performs better than ordinary bump hunting in responder identification. And since stabilized bump hunting requires categorized factors, in order to compare the procedures, we need to consider the regression tree model with categorized factors. For the rest of this chapter, when not otherwise specified, "tree model" would denote the categorized factor tree model from section 10.4.3 and "bump model" the stabilized bump hunting model from section 10.5.

Both the tree and the bump models are hierarchical, so that the first few split nodes of the tree contain predictive factors, which are comparable in their performance to the factors in the first few borders of the first box in the bump model (the main difference, of course, is the type of interaction between the factors). The first level of the regression tree model contains the predictive factor ONSET. The second level contains AGE and DIABETES. ONSET is the most important predictor in the bump model as well, as it defines the first border of the first box. AGE comes in second, DIABETES is the first border of the second box. So we can find the same three factors among the most important predictive factors in both models. Yet the models identify different groups of positive and negative responders. Indeed, further comparison shows that the tree model is much more complicated than the bump model. Notice also that the difference between end nodes of the tree containing positive responders and ones containing negative responders is often just one or two predictors. For example, patients with high onset, diabetes and on beta-blockers are classified as positive responders, whereas patients with high onset, diabetes, off beta-blockers, and with NYHA = 1 are classified as negative responders. The more complicated such models are, the more difficult it is to judge the correctness of such differences clinically.

Let us now compare the groups of patients identified as positive and negative responders in both models. We would be interested in the Amiodarone arm only. Table 10.8 gives a cross-tabulation of the 577 patients as classified in responder and non-responder groups with the help of the regression tree and bump models. A complete mismatch in the classification scheme occurred for a total of 13 patients. Their residuals are plotted in figure 10.15. Only three patients who were identified as positive responders using the bump model were considered to be negative responders when using the tree model. Ten negative responders in the bump model were considered to be positive responders by the tree model.

Table 10.8: *Identified responders in the algorithm with categorized factors regression tree and stabilized bump hunting predictive models. Table cells represent number of patients in the Amiodarone arm. Please refer to table 10.7 and figures 10.7 & 10.8 as well.*

		TREE			Total
		+ responders (-1)	non-responders (0)	- responders (1)	
BUMP	+ responders (-1)	27	47	3	77
	non-responders (0)	35	326	20	381
	- responders (1)	10	88	21	119
Total		72	461	44	577

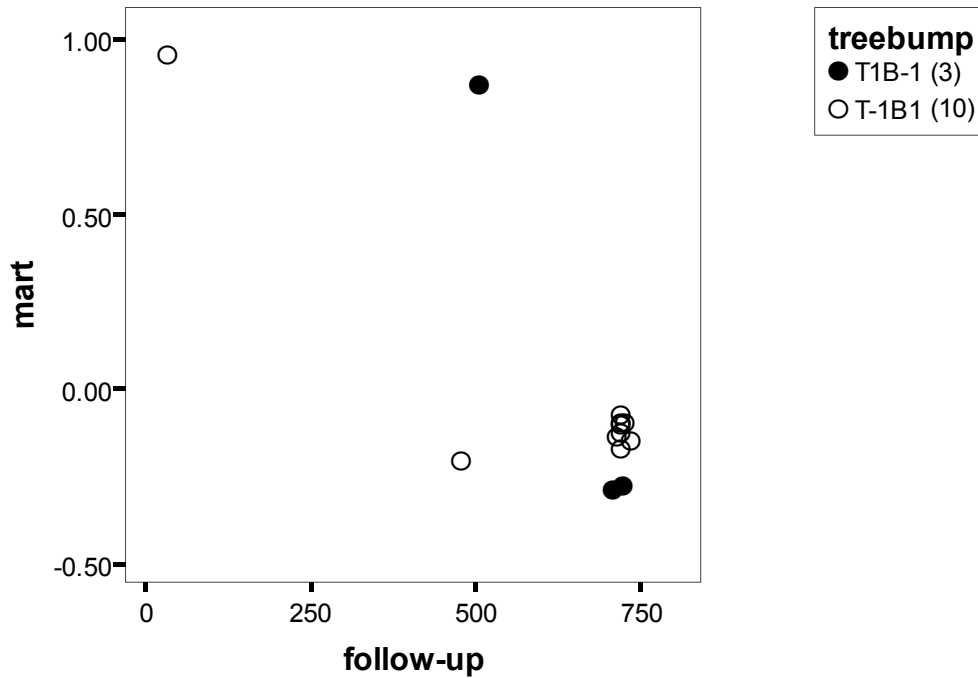


Figure 10.15: *Martingale residuals of patients identified to be positive responders in one model and negative responders in the other and vice versa: T = Tree model, B = Bump model, -1 = + responders, 1 = - responders.*

It is more interesting to look at patients who were identified as positive or negative responders by one model only. In table 10.8, the number of patients who were identified as positive responders in one model only are printed in bold; ones who were identified as negative responders in one model only are in shaded boxes. Figure 10.16 shows four panels, corresponding to the four possible mismatches by the models:

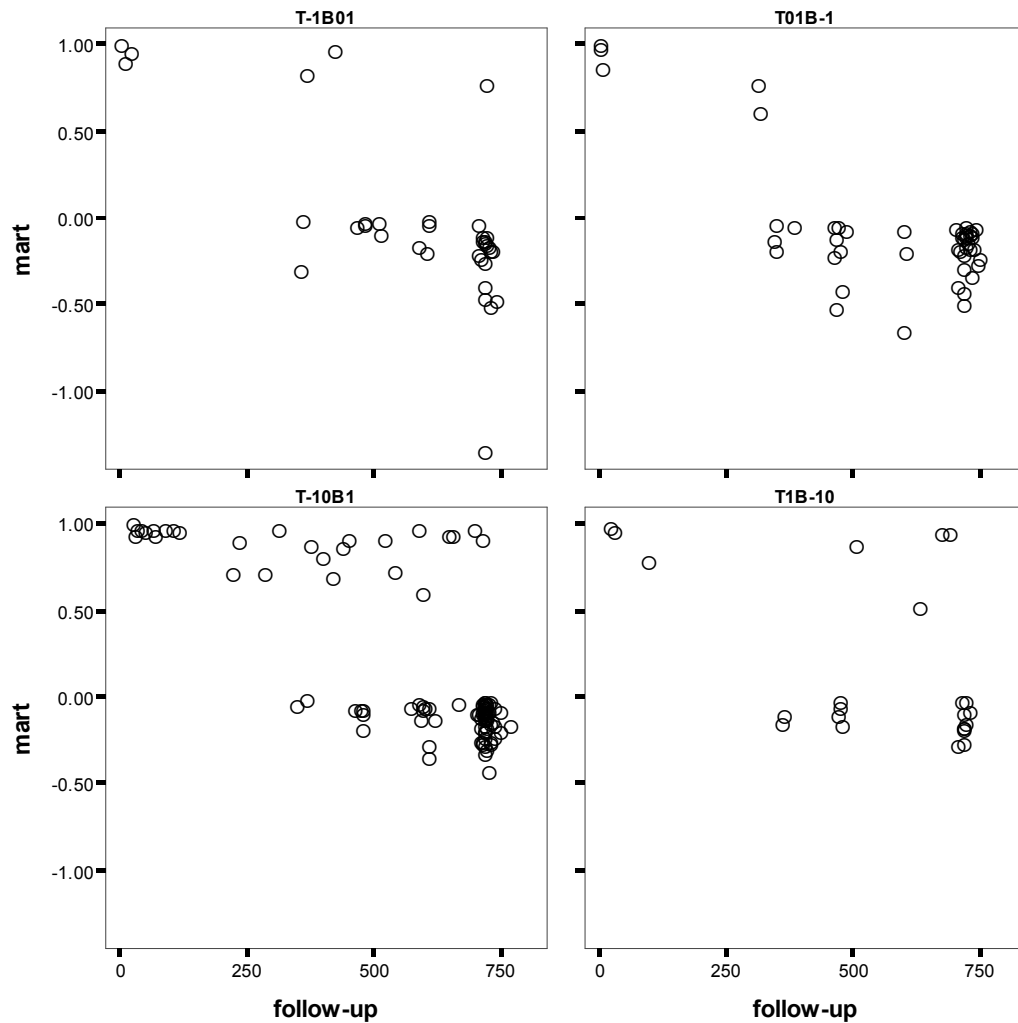


Figure 10.16: *Martingale residuals of patients identified to be positive or negative responders by one model only:*

Panel 1 (T-1B01) contains the 45 patients who were considered to be responders in the tree model only.

Panel 2 (T01B-1) contains 50 patients classified as responders by the bump model only.

Panel 3 (T-10B1) contains 98 patients who were non-responders in the bump model only.

Panel 4 (T1B-10) contains 23 patients who were non-responders in the tree model only.

T = Tree model, B = Bump model, -1 = + responders, 1 = - responders, 0 = non-responders.

Panels 1 & 4 show differences of the bump model with respect to the tree model (notice: fewer patients). Panels 2 & 3 show differences of the tree model with respect to the bump model (notice: more patients). Chapter 9 showed that the bump model should deliver better results than the tree model. To see this, one should not only be looking at the misclassification rate with respect to the other model, but also at the size of the misclassified residuals (figure 10.16) as well as the survival curves for the four panels of figure 10.16.

Consider the misclassified residuals and the type of error which is likely to have occurred, assuming that patients with large positive residuals are likely to be negative responders and patients with large negative residuals are likely to be positive responders. Panels 3 & 4 give insight to model performance with respect to the most dramatic error which can be made: failing to identify negative responders, which results in treating patients with medication, which is harmful for them (kills them). Panel 3 shows the martingale residuals of patients who were not identified as negative responders by the tree model. Concentrating on the positive residuals (patients who died), we notice that they all have values above .5. When comparing the entire group of patient in this panel to the corresponding group in the placebo arm, the survival curves show difference at the .1 level ($p(\text{LR}) = .0986$). Those patients with large positive residuals are likely to belong to the negative responder group, as recognized in the bump model. Panel 4 shows the same for the bump model. Notice, much fewer patients were misclassified with respect to the other model than in panel 3 and only seven had positive residuals, i.e. are likely negative responders and were not chosen as such by the bump model. Panels 1 & 2 give information on the less crucial error, which can be made: failing to identify positive responders, i.e. denying medication to patients, which would actually help them (improves survival). Very few possible errors were made here, which appear on the plots as large negative residuals (e.g. $< -.5$). The performance of the two models in this case is comparable.

In general, the bump model had less (presumably) misclassified patients than the tree model, as it was to be expected after the simulation study in chapter 9.

A note on software and algorithmic implementation:

The above analysis was done with the help of SPSS 10.0, S-PLUS 4.5 (for Windows and for Unix), functions from Becker's bump hunting software (1999), as well as some S functions, written to automate the bootstrapping part of the stabilized bump hunting algorithm. All self generated code is given in Appendix C.

CONCLUSIONS

Summary

Clinical trials often judge the efficacy of a new treatment by comparing the outcome (survival patterns) of patients randomly assigned to undergo a new or a standard treatment. Usually, the entire groups are analyzed, although it is well known that certain subgroups of patients react differently to the new treatment than others. Some patients taking the new treatment might benefit from it (the positive responders) while others may be harmed by it (the negative responders). The topic of this thesis is extraction of such special subgroups of patients, based on finding the so called predictive factors, which describe survival differences solely due to the new treatment.

The thesis gives an overview of the techniques used up to now for responder identification and proposes a new method for systematic search for responders. The responder identification method consists of the following three steps:

1. Identification of "prognostic" factors (e.g. via Cox-PH model on the standard treatment arm). Notice, those factors are prognostic in the classical sense only if the study was performed with placebo, not standard treatment arm.
2. Identification of patients in the new treatment arm, who's survival time is badly estimated by the prognostic model (e.g. via search for outliers in the deviance or martingale residuals)
3. Identification of predictive factors, which describe common features of the patients with residual outliers, namely the positive and negative responders (e.g. via regression tree or bump hunting analysis, or via the suggested stabilized bump hunting procedure)

The basic responder identification method was developed for analysis of clinical trial data, in which no difference in survival between the new and the classical treatment

groups is present. Slight changes to the method were discussed for application on data, which does show initial difference in survival between the two treatment groups. Several variations of the basic responder identification method were proposed and compared in a simulation study.

In the search for predictive factors, one can apply martingale or deviance residuals to the prognostic model as a response variable in a regression tree, bump hunting, or the proposed stabilized bump hunting analysis. The simulation study showed that martingale residuals, combined with the stabilized bump hunting procedure are most suitable for responder identification. This variation of the suggested procedure has power of 99% (i.e. recognized the correct positive and negative responder groups 99% of the time).

Some versions of the proposed responder identification method were also applied on a "real life" data set – the European Myocardial Infarction Amiodarone Trial (EMIAT) and the identified positive and negative responder groups were compared.

All versions of the proposed responder identification algorithm, and especially the one employing stabilized bump hunting with martingale residuals as response, perform better than the method available up to now – Cox-PH model with treatment interactions. This was shown in the simulation study and in the analysis of the EMIAT data. The better performance of the new method is due to the fact that it recognizes interactions of higher order between covariates much easier than the Cox-PH model does.

Outlook

Fully automated implementation of the six different versions of the responder identification algorithm were done only for the simulation study, where the number and type of factors in the data were set. Responder analysis of the EMIAT data was done with semi-automated implementation, which allows more flexibility in the predictive model building process, but also slows down the analysis. If the best version of the responder identification algorithm, stabilized bump hunting with martingale residuals as

response, is to be used on a regular basis, a computer program is needed, which fully automates the procedure, allowing for different number and type of factors, different size of the data set, and flexibility in choosing the stopping criteria for model growth.

The responder identification procedure was created for and tested on data sets with no survival difference between the new and the classical treatment groups. A future software product for responder analysis should allow for data with initial survival difference as well. The performance of the proposed responder identification method on such data may be tested in a simulation study.

So far, it was also assumed that none of the prognostic and predictive factors vary with time. A new study can be done, which extends responder identification to data with time varying effects as well.

APPENDIX A: PROOFS AND EXAMPLES

1. Reduction of $\hat{M}_i(t)$ to \hat{M}_i

In chapter 3.3 we defined Martingale residuals as follows (3.9):

$$\hat{M}_i(t) = N_i(t) - \int_0^t Y_i(s) \cdot e^{\hat{\beta}' \cdot Z_i(s)} d\hat{\Lambda}_0(s)$$

using Stieltjes Integral

$$= N_i(t) - \int_0^t Y_i(s) \cdot e^{\hat{\beta}' \cdot Z_i(s)} \cdot \hat{\lambda}_0(s) ds$$

For a right censored data time-constant model, such as Cox-PH:

$$\hat{M}_i = \delta_i - e^{\hat{\beta}' \cdot Z_i} \cdot \int_0^{\tau_i} Y_i(s) \cdot \hat{\lambda}_0(s) ds$$

where:

τ_i is the observation time for subject i and

δ_i is the final status for subject i .

Let us consider the integral part of the equation above and let

$$f_i(x) = Y_i(x) \cdot \hat{\lambda}_0(x)$$

Then:

$$\begin{aligned}
\int_0^{\tau_i} Y_i(s) \cdot \hat{\lambda}_0(s) ds &= \int_0^{\tau_i} f_i(s) ds \\
&= F_i(\tau_i) - F_i(0) \quad , \text{ which can be approximated by} \\
&= \sum_{t \leq \tau_i} f_i(t) - \sum_{t \leq 0} f_i(t) \\
&= \sum_{t \leq \tau_i} Y_i(t) \cdot \hat{\lambda}_0(t) - \underbrace{Y_i(0) \cdot \hat{\lambda}_0(0)}_{=0} \quad \text{as } t \in [0, \tau_i] \\
&= \sum_{t^c \leq \tau_i} 0 \cdot \hat{\lambda}_0(t) + \sum_{t^d \leq \tau_i} 1 \cdot \hat{\lambda}_0(t) = \hat{\Lambda}_0(\tau_i)
\end{aligned}$$

where t^c denotes censoring times and t^d – observed failure times. Now we can return to \hat{M}_i where for the Cox-PH model we have the following:

$$\hat{M}_i = \delta_i - \hat{\Lambda}_0(\tau_i) \cdot e^{\hat{\beta}' \cdot Z_i} = \delta_i - \hat{\Lambda}_i(\tau_i, Z_i)$$

■

2. Deviance residuals examined by cases

Deviance residuals, as defined in 3.16, can be combined with the definition of Martingale residuals (3.10) and transformed the following way:

$$\begin{aligned}
d_i &= \text{sgn}(\hat{M}_i) \cdot \sqrt{-2 \cdot [\hat{M}_i + \delta_i \cdot \ln(\delta_i - \hat{M}_i)]} \\
&= \text{sgn}(\delta_i - \hat{\Lambda}_i) \cdot \sqrt{-2 \cdot [\delta_i - \hat{\Lambda}_i + \delta_i \cdot \ln(\delta_i - \delta_i + \hat{\Lambda}_i)]} \\
&= \text{sgn}(\delta_i - \hat{\Lambda}_i) \cdot \sqrt{2 \cdot (\hat{\Lambda}_i - \delta_i - \delta_i \cdot \ln \hat{\Lambda}_i)}
\end{aligned}$$

Case $\delta_i = 0$:

$$d_i = \text{sgn}(-\hat{\Lambda}_i) \sqrt{2\hat{\Lambda}_i} = -\sqrt{2\hat{\Lambda}_i}, \quad \hat{\Lambda}_i \in [0, \infty)$$

$$d_i = 0 \quad \text{if} \quad \hat{\Lambda}_i = 0$$

$$d_i > 0 \quad - \quad \text{impossible in this case}$$

$$d_i = -2 \quad \text{if} \quad \hat{\Lambda}_i = 2$$

and in general (see figure #):

$$d_i < 0 \quad \text{if} \quad \hat{\Lambda}_i > 0$$

Case $\delta_i = 1$:

$$d_i = \text{sgn}(1 - \hat{\Lambda}_i) \sqrt{2(\hat{\Lambda}_i - 1 - \ln \hat{\Lambda}_i)}, \quad \hat{\Lambda}_i \in (0, \infty)$$

$$d_i = 0 \quad \text{if} \quad 2(\hat{\Lambda}_i - 1 - \ln \hat{\Lambda}_i) = 0$$

$$\Updownarrow$$

$$\hat{\Lambda}_i = 1$$

$$d_i = 2 \quad \text{if} \quad \hat{\Lambda}_i < 1 \quad \& \quad \sqrt{2(\hat{\Lambda}_i - 1 - \ln \hat{\Lambda}_i)} = 2$$

$$\Updownarrow$$

$$\hat{\Lambda}_i - 3 - \ln \hat{\Lambda}_i = 0$$

$$\Updownarrow^*$$

$$\hat{\Lambda}_i \approx 0.0525$$

$$d_i = -2 \quad \text{if} \quad \hat{\Lambda}_i > 1 \quad \& \quad -\sqrt{2(\hat{\Lambda}_i - 1 - \ln \hat{\Lambda}_i)} = -2$$

$$\Updownarrow$$

$$\hat{\Lambda}_i - 3 - \ln \hat{\Lambda}_i = 0$$

$$\Updownarrow^*$$

$$\hat{\Lambda}_i \approx 4.5052$$

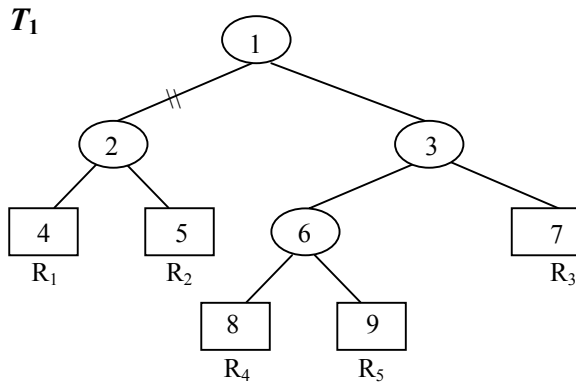
and in general:

$$d_i < 0 \quad \text{if} \quad \hat{\Lambda}_i > 1$$

$$d_i > 0 \quad \text{if} \quad 0 < \hat{\Lambda}_i < 1$$

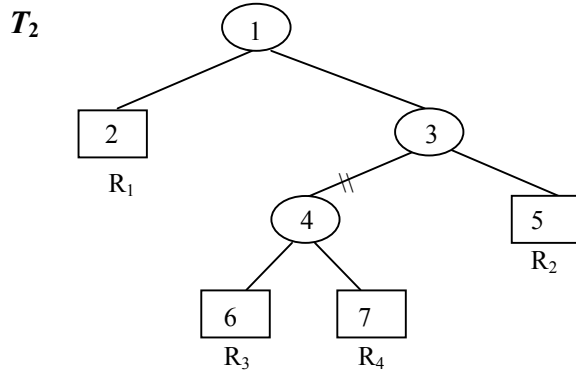
* solutions were found using Maple 6 and the function "solve".

3. Pruning example to chapter 5



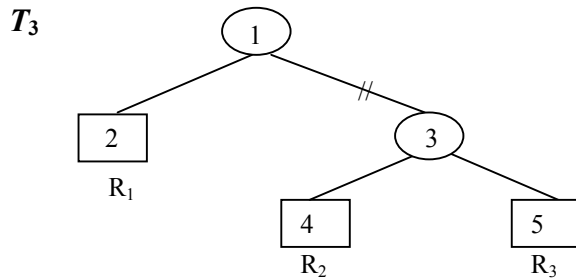
$$|T_1| = 5$$

$$C_\alpha(T_1) = \sum_{m=1}^5 \sum_{x_i \in R_m} (y_i - \hat{c}_m)^2 - 5\alpha$$



$$|T_2| = 4$$

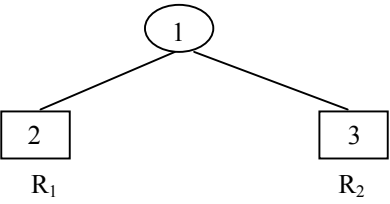
$$C_\alpha(T_2) = \sum_{m=1}^4 \sum_{x_i \in R_m} (y_i - \hat{c}_m)^2 - 4\alpha$$



$$|T_3| = 3$$

$$C_\alpha(T_3) = \sum_{m=1}^3 \sum_{x_i \in R_m} (y_i - \hat{c}_m)^2 - 3\alpha$$

T_4



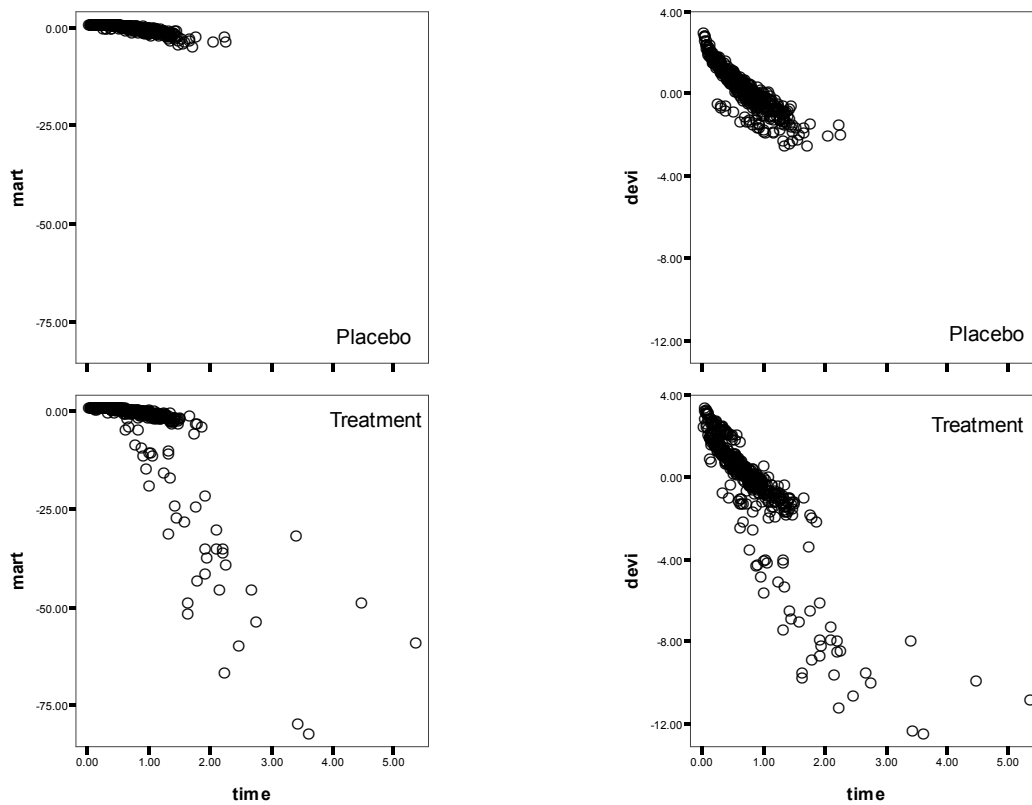
$$|T_4| = 2$$

$$C_\alpha(T_4) = \sum_{m=1}^2 \sum_{x_i \in R_m} (y_i - \hat{c}_m)^2 - 2\alpha$$

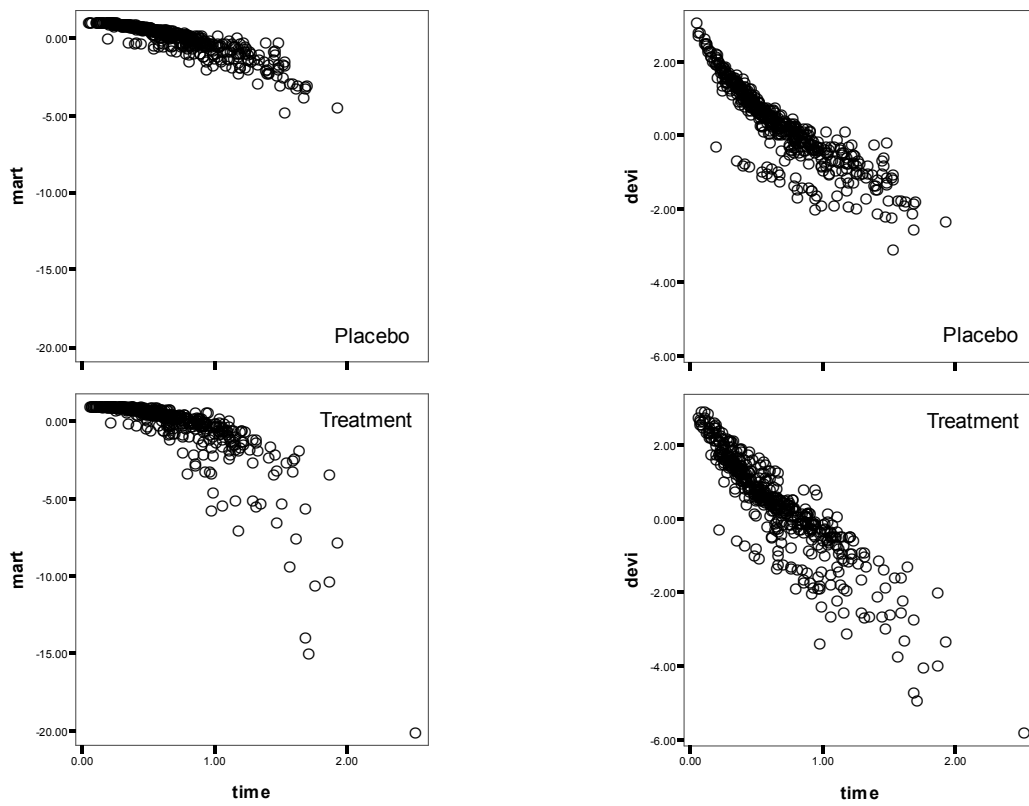
APPENDIX B: SIMULATION STUDY PLOTS

1. Simulation study: scatter plots of the residuals

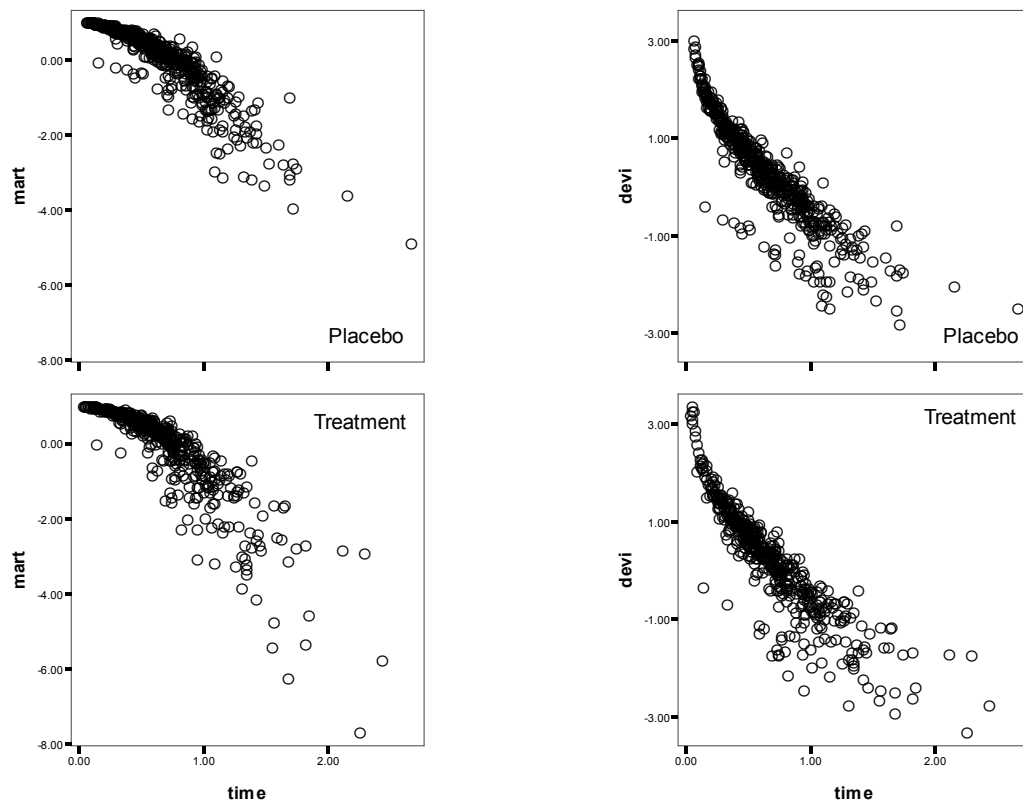
Simulation group 1 ($c_{\min} = -2$, $c_{\max} = 2$, censored = 10%):



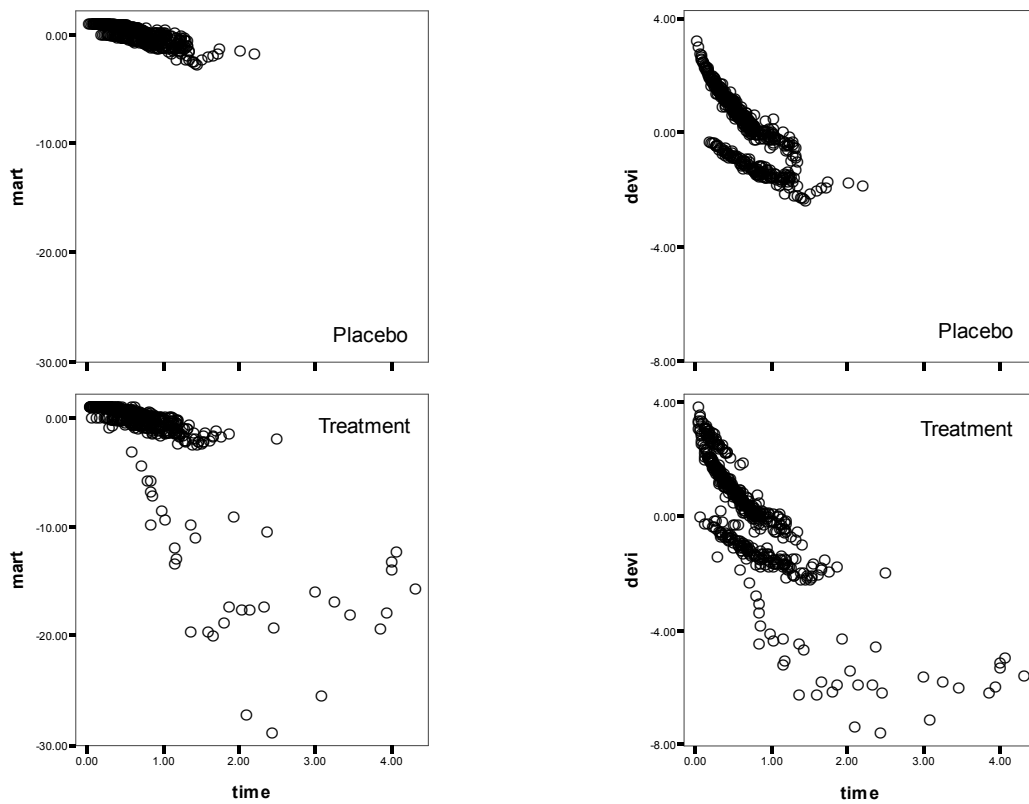
Simulation group 2 ($c_{\min} = -1$, $c_{\max} = 1$, censored = 10%):



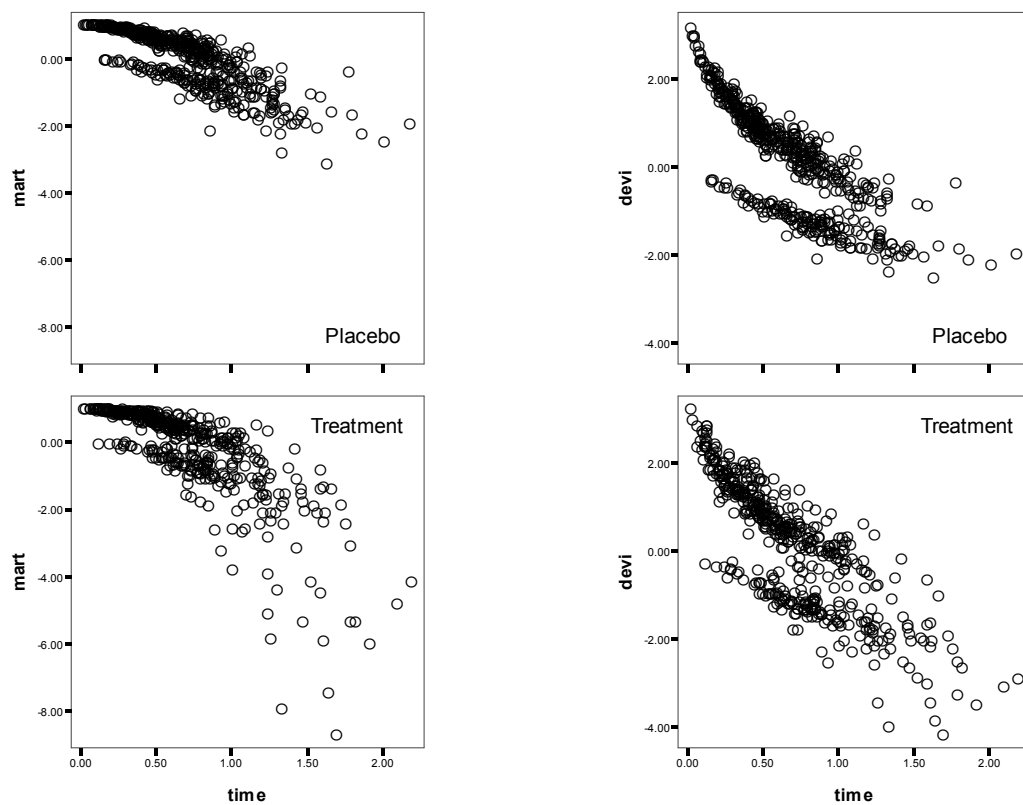
Simulation group 3 ($c_{\min} = -0.5$, $c_{\max} = 0.25$, censored = 10%):



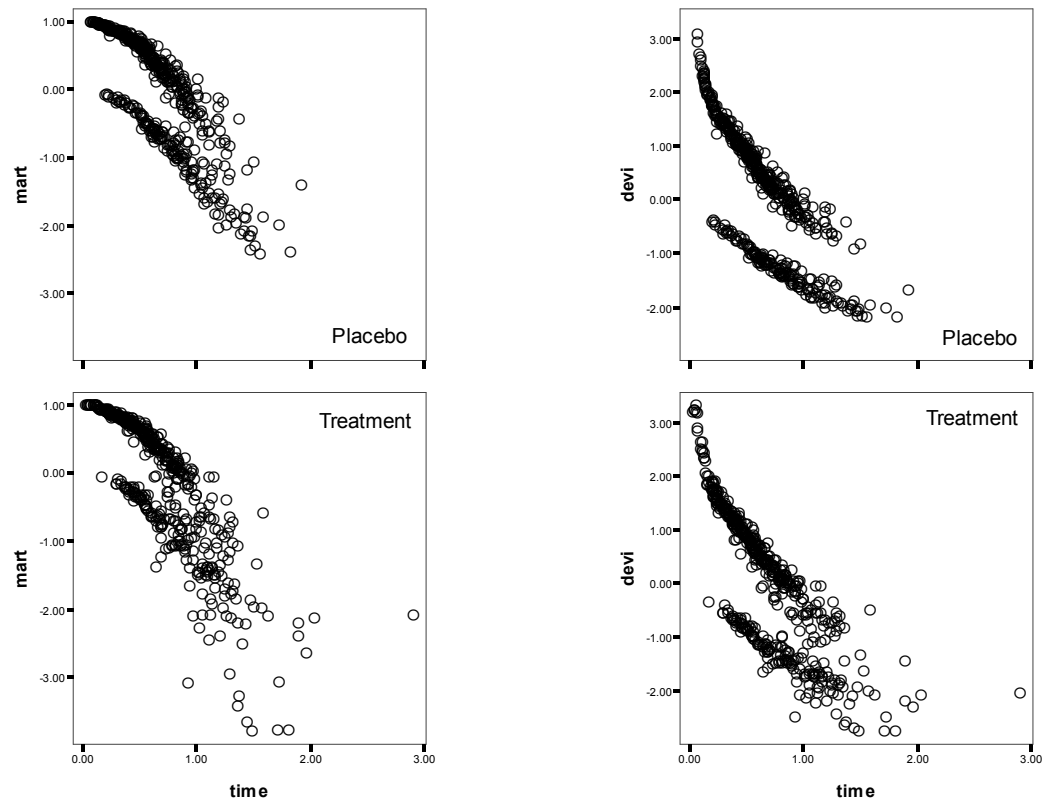
Simulation group 4 ($c_{\min} = -2$, $c_{\max} = 2$, censored = 30%):



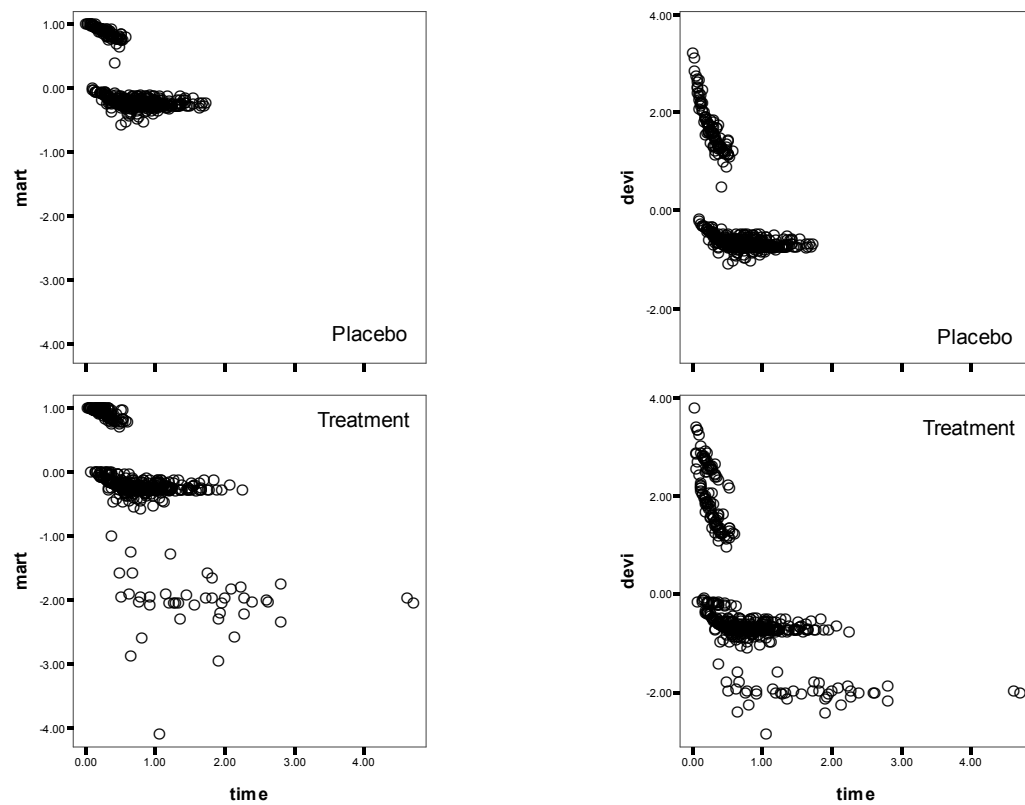
Simulation group 5 ($c_{\min} = -1$, $c_{\max} = 1$, censored = 30%):



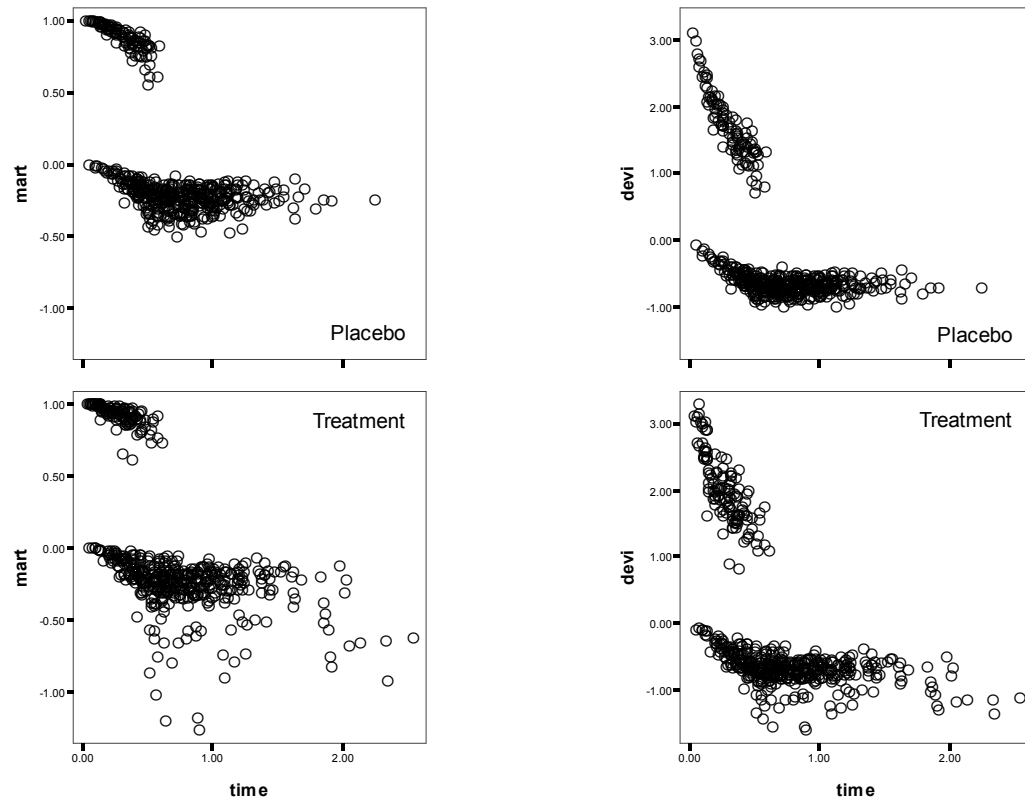
Simulation group 6 ($c_{\min} = -0.5$, $c_{\max} = 0.25$, censored = 30%):



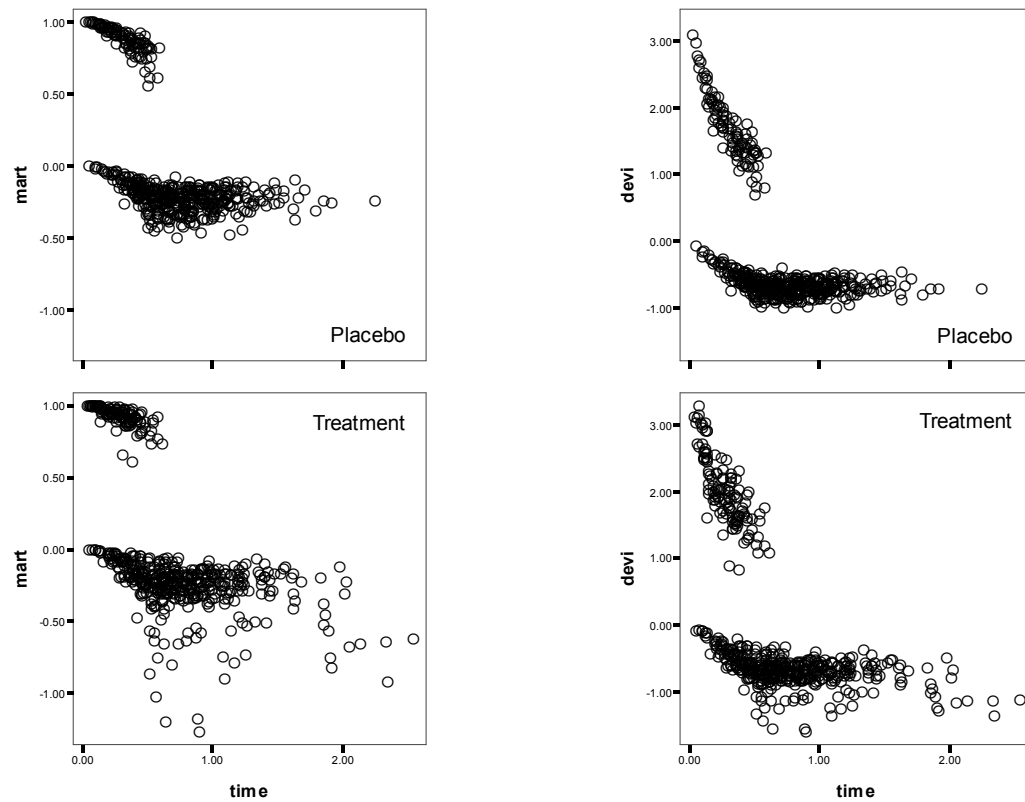
Simulation group 7 ($c_{\min} = -2$, $c_{\max} = 2$, censored = 70%):



Simulation group 8 ($c_{\min} = -1$, $c_{\max} = 1$, censored = 70%):



Simulation group 9 ($c_{\min} = -0.5$, $c_{\max} = 0.25$, censored = 70%):



APPENDIX C: ALGORITHMIC IMPLEMENTATION IN S

1. Functions needed for simulations with bump hunting (on UNIX)

```
".boot.sim" <- function(nboot=10, coeffmin=1, coeffmax=-2, cens.max=11, beta=.1)
{
  rownames <- c("x10", "x11", "x20", "x21", "x22", "x30", "x31", "x40", "x41", "x50", "x51",
"x60", "x61")
  output <- rep(0,4)
  simdata <- .data.sim(coeffmin=coeffmin, coeffmax=coeffmax, cens.max=11)
  martdata <- .model.fit(simdata)
  prepdata.max <- .box.prep(martdata)
  prepdata.min <- prepdata.max
  minbox <- .boxes(train.data = prepdata.min, type = rep(1, 6), nboxes = 1, maxi = F, beta =
beta, pasting = F, peel.crit = 2, output = F)
  min.rank <- .border.ranking(minbox, crit.valid = F)
  maxbox <- .boxes(train.data = prepdata.max, type = rep(1, 6), nboxes = 1, maxi = T, beta =
beta, pasting = F, peel.crit = 2, output = F)
  max.rank <- .border.ranking(maxbox, crit.valid = F)

  # min original bump hunting
  onm <- dimnames(min.rank[[4]])[[1]]
  orm <- min.rank[[4]][,3]
  n <- length(onm)
  if(onm[n]=="x4" && onm[n-1]=="x5" && onm[n-2]=="x6" && orm[n]=="= 0" && orm[n-1]=="=
0" && orm[n-2]=="= 0") output[1] <- 1
  if(onm[n]=="x4" && onm[n-1]=="x6" && onm[n-2]=="x5" && orm[n]=="= 0" && orm[n-1]=="=
0" && orm[n-2]=="= 0") output[1] <- 1
  if(onm[n]=="x5" && onm[n-1]=="x6" && onm[n-2]=="x4" && orm[n]=="= 0" && orm[n-1]=="=
0" && orm[n-2]=="= 0") output[1] <- 1
  if(onm[n]=="x5" && onm[n-1]=="x4" && onm[n-2]=="x6" && orm[n]=="= 0" && orm[n-1]=="=
0" && orm[n-2]=="= 0") output[1] <- 1
  if(onm[n]=="x6" && onm[n-1]=="x4" && onm[n-2]=="x5" && orm[n]=="= 0" && orm[n-1]=="=
0" && orm[n-2]=="= 0") output[1] <- 1
  if(onm[n]=="x6" && onm[n-1]=="x5" && onm[n-2]=="x4" && orm[n]=="= 0" && orm[n-1]=="=
0" && orm[n-2]=="= 0") output[1] <- 1

  # max original BH
  onx <- dimnames(max.rank[[4]])[[1]]
  orx <- max.rank[[4]][,3]
  n <- length(onx)
```

```

if(onx[n]=="x5" && onx[n-1]=="x2" && onx[n-2]=="x2" && orx[n]=="0" && orx[n-1]=="0"
&& orx[n-2]=="1") output[2] <- 1
if(onx[n]=="x5" && onx[n-1]=="x2" && onx[n-2]=="x2" && orx[n]=="0" && orx[n-1]=="1"
&& orx[n-2]=="0") output[2] <- 1
if(onx[n]=="x2" && onx[n-1]=="x2" && onx[n-2]=="x5" && orx[n]=="0" && orx[n-1]=="1"
&& orx[n-2]=="0") output[2] <- 1
if(onx[n]=="x2" && onx[n-1]=="x5" && onx[n-2]=="x2" && orx[n]=="0" && orx[n-1]=="0"
&& orx[n-2]=="1") output[2] <- 1
if(onx[n]=="x2" && onx[n-1]=="x5" && onx[n-2]=="x2" && orx[n]=="1" && orx[n-1]=="0"
&& orx[n-2]=="0") output[2] <- 1
if(onx[n]=="x2" && onx[n-1]=="x2" && onx[n-2]=="x5" && orx[n]=="1" && orx[n-1]=="0"
&& orx[n-2]=="0") output[2] <- 1

```

```

# initiate resulting vectors
orignam.max <- rep("0", 14)
origrestr.max <- rep("0", 14)
orignam.min <- rep("0", 14)
origrestr.min <- rep("0", 14)
boxmax <- rep("0", 14)
boxmin <- rep("0", 14)
res.supp.max <- c(1,rep(0, 13))
res.supp.min <- c(1,rep(0, 13))
res.mean.max <- rep(0, 14)
res.mean.max[1] <- mean(prepdata.max$mart)
res.mean.min <- rep(0, 14)
res.mean.min[1] <- res.mean.max[1]
res.lr.max <- rep(0, 14)
lrmax <- logrank(S=Surv(martdata$time, martdata$death), group=martdata$treat)
res.lr.max[1] <- lrmax$pval
res.lr.min <- rep(0, 14)
res.lr.min[1] <- res.lr.max[1]
treat.num <- dim(prepdata.max)[1]

```

```

# constructing boxmax
cat("maxbox", "\n")
nbord <- 1
spri <- F
while(spri == F)
{
  outmat.max <- matrix(0, 13, nboot+1, dimnames=list(rownames, NULL))
  oresult <- .evaluate(prepdata.max, maxi=T, beta=.05)
  op <- dim(oresult)[4][1]
  orignam.max[nbord] <- dimnames(oresult)[4][1][op]
  origrestr.max[nbord] <- oresult[4][op,3]
  j <- 1 # bootstrap sample number
  while(j < nboot+1) #bootstrap samples
  {

```

```

length <- dim(prepdata.max)[1]
n <- c(1:length)
samp <- sample(n, replace=T)
sampdata <- prepdata.max[samp,]
names(sampdata) <- names(prepdata.max)
result <- .evaluate(sampdata, maxi=T, beta=.05)
p <- dim(result[[4]])[1]
varnam <- dimnames(result[[4]])[[1]][p]
restrict <- result[[4]][p,3]
outmat.max <- .fals.funk(j=j, varnam=varnam, restrict=restrict, outmat=outmat.max)
j <- j+1
}
for(i in 1:13)
  outmat.max[i,nboot+1] <- sum(outmat.max[i,])
cat(outmat.max[,nboot+1], "\n")
maximal <- max(outmat.max[,nboot+1])

# if two borders are maximal, take the one with the bigger support
support <- rep(0,13)
for(i in 1:13)
  if(outmat.max[i,nboot+1]==maximal)
  {
    maxrow <- rownames[i]
    tempo.max <- .restrict(prepdata=prepdata.max, maxrow=maxrow)
    support[i] <- dim(tempo.max)[1]
  }
maxi.supp <- max(support)
for(i in 1:13)
  if(support[i] == maxi.supp) index <- i
maxrow <- rownames[index]
prepdata.max <- .restrict(prepdata=prepdata.max, maxrow=maxrow)
boxmax[nbord+1] <- maxrow

# update loop parameter
nbord <- nbord+1
supp.max <- dim(prepdata.max)[1]
res.supp.max[nbord] <- supp.max/treat.num
res.mean.max[nbord] <- mean(prepdata.max$mart)
boxdata.max <- .box.restrict(box=boxmax, martdata)
lrmax <- .logrank(S=Surv(boxdata.max$time, boxdata.max$death),
group=boxdata.max$treat)
res.lr.max[nbord] <- lrmax$pval

# stopping criteria
if(res.supp.max[nbord] < beta)
  spri <- T
}

```

```

# constructing minbox
cat("minbox", "\n")
nbord <- 1
spri <- F
while(spri == F)
{
  outmat.min <- matrix(0, 13, nboot+1, dimnames=list(rownames, NULL))
  oresult <- .evaluate(prepdata.min, maxi=F, beta=.05)
  op <- dim(oreresult[[4]])[1]
  orignam.min[nbord] <- dimnames(oreresult[[4]] [[1]])[op]
  origrestr.min[nbord] <- oresult[[4]][op,3]
  j<- 1      # bootstrap sample number
  while(j < nboot+1)      #bootstrap samples
  {
    length <- dim(prepdata.min)[1]
    n <- c(1:length)
    samp <- sample(n, replace=T)
    sampdata <- prepdata.min[samp,]
    names(sampdata) <- names(prepdata.min)
    result <- .evaluate(sampdata, maxi=F, beta=.05)
    p <- dim(result[[4]])[1]
    varnam <- dimnames(result[[4]] [[1]])[p]
    restrict <- result[[4]][p,3]
    outmat.min <- .fals.funk(j=j, varnam=varnam, restrict=restrict, outmat=outmat.min)
    j <- j+1
  }

  for(i in 1:13)
    outmat.min[i,nboot+1] <- sum(outmat.min[i,])
  cat(outmat.min[,nboot+1], "\n")
  maximal <- max(outmat.min[,nboot+1])

  # if two borders are maximal, take the one with the bigger support
  support <- rep(0,13)
  for(i in 1:13)
    if(outmat.min[i,nboot+1]==maximal)
    {
      maxrow <- rownames[i]
      tempo.min <- .restrict(prepdata=prepdata.min, maxrow=maxrow)
      support[i] <- dim(tempo.min)[1]
    }
  maxi.supp <- max(support)
  for(i in 1:13)
    if(support[i] == maxi.supp) index <- i
  maxrow <- rownames[index]
  prepdata.min <- .restrict(prepdata=prepdata.min, maxrow=maxrow)

```

```

boxmin[nbord+1] <- maxrow

# update loop parameter
nbord <- nbord+1
supp.min <- dim(prepdata.min)[1]
res.supp.min[nbord] <- supp.min/treat.num
res.mean.min[nbord] <- mean(prepdata.min$mart)
boxdata.min <- .box.restrict(box=boxmin, martdata)
lrmin <- .logrank(S=Surv(boxdata.min$time, boxdata.min$death),
group=boxdata.min$treat)
res.lr.min[nbord] <- lrmin$pval

# stopping criteria
if(res.supp.min[nbord] < beta)
  spri <- T
}

# cutting all zeros
spri <- F
i.max <- 2
while(spri == F)
{
  if(boxmax[i.max] == "0")
    spri <- T
  else
    i.max <- i.max + 1
}
spri <- F
i.min <- 2
while(spri == F)
{
  if(boxmin[i.min] == "0")
    spri <- T
  else
    i.min <- i.min + 1
}
boxmax <- boxmax[1:i.max-1]
boxmin <- boxmin[1:i.min-1]
res.supp.max <- res.supp.max[1:i.max-1]
res.supp.min <- res.supp.min[1:i.min-1]
res.mean.max <- res.mean.max[1:i.max-1]
res.mean.min <- res.mean.min[1:i.min-1]
res.lr.max <- res.lr.max[1:i.max-1]
res.lr.min <- res.lr.min[1:i.min-1]
if(boxmin[2]=="x40" && boxmin[3]=="x50" && boxmin[4]=="x60")
  output[3] <- 1
if(boxmin[2]=="x40" && boxmin[3]=="x60" && boxmin[4]=="x50")

```

```

    output[3] <- 1
    if(boxmin[2]=="x50" && boxmin[3]=="x60" && boxmin[4]=="x40")
      output[3] <- 1
    if(boxmin[2]=="x50" && boxmin[3]=="x40" && boxmin[4]=="x60")
      output[3] <- 1
    if(boxmin[2]=="x60" && boxmin[3]=="x40" && boxmin[4]=="x50")
      output[3] <- 1
    if(boxmin[2]=="x60" && boxmin[3]=="x50" && boxmin[4]=="x40")
      output[3] <- 1
    if(boxmax[2]=="x50" && boxmax[3]=="x20" && boxmax[4]=="x21")
      output[4] <- 1
    if(boxmax[2]=="x50" && boxmax[3]=="x21" && boxmax[4]=="x20")
      output[4] <- 1
    if(boxmax[2]=="x20" && boxmax[3]=="x21" && boxmax[4]=="x50")
      output[4] <- 1
    if(boxmax[2]=="x20" && boxmax[3]=="x50" && boxmax[4]=="x21")
      output[4] <- 1
    if(boxmax[2]=="x21" && boxmax[3]=="x50" && boxmax[4]=="x20")
      output[4] <- 1
    if(boxmax[2]=="x21" && boxmax[3]=="x20" && boxmax[4]=="x50")
      output[4] <- 1

    return(output, boxmin, boxmax, min.rank[[4]][,3], max.rank[[4]][,3])
  }

".box.prep"<-function(daten)
{
  # prepares data from .model.fit for use in .boxes
  treat <- daten[, 9]
  simtreat <- daten[treat == 1, ]
  names(simtreat) <- names(daten)
  mart <- simtreat[, c(14, 2:8)]
  mart <- mart[, -4]
  m <- mean(mart[,4])
  for(i in 1:dim(mart)[1])
    if(mart[i,4] <= m) mart[i,4] <- 0
    else mart[i,4] <- 1
  return(mart)
}

".box.prep.devi"<-function(daten)
{
  # prepares data from .model.fit for use in .boxes
  treat <- daten[, 9]
  simtreat <- daten[treat == 1, ]
  names(simtreat) <- names(daten)
  devi <- simtreat[, c(17, 2:8)]

```

```

devi <- devi[, -4]
m <- mean(devi[, 4])
for(i in 1:dim(devi)[1])
  if(devi[i, 4] <= m) devi[i, 4] <- 0 else devi[i, 4] <- 1
return(devi)
}

".data.sim"<-function(coeffmin = 1, coeffmax = -2, cens.max = 10)
{
  n <- c(1:1000)
  x1 <- rbinom(1000, 1, 0.5)
  x2 <- rep(0, 1000)
  tempo <- runif(1000, min = 0, max = 1)
  for(i in 1:1000)
    if(tempo[i] > 0.3333) x2[i] <- 1
  for(i in 1:1000)
    if(tempo[i] > 0.6667) x2[i] <- 2
  ind2x2 <- rep(0, 1000)
  for(i in 1:1000)
    if(x2[i] == 2) ind2x2[i] <- 1
  x3 <- rnorm(1000, mean = 5, sd = 2)
  x4 <- rbinom(1000, 1, 0.5)
  x5 <- rbinom(1000, 1, 0.5)
  x6 <- rbinom(1000, 1, 0.5)
  treat <- rbinom(1000, 1, 0.5)
  rr <- -log(3) * x1 + (2 * log(3) * x1 * x3)/10 - (log(3) * x3)/10 + coeffmax * ind2x2 * x5 *
  treat + coeffmin * x4 * x5 * x6 * treat
  lambda <- exp(rr)
  simdata <- cbind(n, x1, x2, ind2x2, x3, x4, x5, x6, treat, rr, lambda)
  simdata <- as.data.frame(simdata)
  zeit <- rep(0, 1000)
  for(i in 1:1000)
    zeit[i] <- rweibull(1, shape = 2, scale = sqrt(lambda[i]))
  censor <- runif(1000, min = 0, max = cens.max)
  death <- rep(0, 1000)
  l <- (zeit <= censor)
  for(i in 1:1000)
    if(l[i] == 1) death[i] <- 1
  simdata <- cbind(simdata, zeit, death)
  names(simdata)[12] <- "time"
  names(simdata)[13] <- "death"
  return(simdata)
}

".evaluate"<-function(boxdata, maxi=T, beta=.05)
{

```

```

# takes data from .box.prep and runs .boxes; uses .express.boxes to see which variables
are in the box
type <- rep(1, 6)
x2 <- boxdata[, 3]
x5 <- boxdata[, 6]
box <- .boxes.vic(train.data = boxdata, type = type, nboxes = 1, maxi = maxi, beta=beta,
pasting = F, peel.crit = 2, output=F)
if(box$flag == 0)
{
  rank.bord <- NA
  cat("no more boxes can be found", "\n")
}
else
{
  descr <- .express.boxes(box$result)
  rank.bord <- .border.ranking(box$result, crit.valid = F)
}
return(rank.bord)
}

".fals.funk" <- function(j, varnam, restrict, outmat)
{
  if((varnam == "x1") && (restrict == "= 0")) outmat[1,j] <- 1
  if((varnam == "x1") && (restrict == "= 1")) outmat[2,j] <- 1
  if((varnam == "x2") && (restrict == "= 0")) outmat[3,j] <- 1
  if((varnam == "x2") && (restrict == "= 1")) outmat[4,j] <- 1
  if((varnam == "x2") && (restrict == "= 2")) outmat[5,j] <- 1
  if((varnam == "x3") && (restrict == "= 0")) outmat[6,j] <- 1
  if((varnam == "x3") && (restrict == "= 1")) outmat[7,j] <- 1
  if((varnam == "x4") && (restrict == "= 0")) outmat[8,j] <- 1
  if((varnam == "x4") && (restrict == "= 1")) outmat[9,j] <- 1
  if((varnam == "x5") && (restrict == "= 0")) outmat[10,j] <- 1
  if((varnam == "x5") && (restrict == "= 1")) outmat[11,j] <- 1
  if((varnam == "x6") && (restrict == "= 0")) outmat[12,j] <- 1

```

```

    if((varnam == "x6") && (restrict == "= 1")) outmat[13,j] <- 1

    return(outmat)
}

".logrank" <- function(S, group)
{
  for(j in 1:length(group))
    if(group[j]==0) group[j]<-2
  y <- S[,1]
  event <- S[,2]
  i <- order(-y)
  y <- y[i]
  event <- event[i]
  group <- group[i]
  x <- cbind(group==1, group==2, (group==1)*event, (group==2)*event)
  s <- rowsum(x, y, F)
  nr1 <- cumsum(s[,1])
  nr2 <- cumsum(s[,2])
  d1 <- s[,3]
  d2 <- s[,4]
  rd <- d1 + d2
  rs <- nr1 + nr2 - rd
  n <- nr1+nr2
  oecum <- d1 - (rd*nr1)/n
  vcum <- (rd*rs*nr1*nr2)/n/n/(n-1)
  chival <- sum(oecum)^2/sum(vcum, na.rm=T)
  pval <- 1-pchisq(chival,df=1)
  return(chival, pval)
}

".model.fit" <- function(daten, coeffmin = 1, coeffmax = -2)
{
  ind <- order(daten$treat, daten$time, rev(daten$death))
  ordtreat <- daten[ind, ]
  names(ordtreat) <- names(daten)
  # omits all (first) rows in the data set for which delta(t1)=0
  while(ordtreat$death[1]==0) ordtreat<-ordtreat[-1,]
  treat <- ordtreat[, 9]
  plac <- ordtreat[treat == 0, ]
  coxmod <- coxph(Surv(time, death) ~ x1 + x3 + x1 * x3, data = plac, method = "breslow", x
= T)
  coxdetail <- coxph.detail(coxmod)
  beta <- as.vector(coxmod$coefficients)
  mart <- coxmod$residuals
  hazard <- plac$death - mart
  plac.linpred <- beta[1] * plac$x1 + beta[2] * plac$x3 + beta[3] * plac$x1 * plac$x3

```

```

basehaz <- hazard/exp(plac.linpred)
len <- length(mart)
m <- dim(ordtreat)[1]
martfull <- c(mart, rep(-999, (m - len)))
hazfull <- c(hazard, rep(-999, (m - len)))
basefull <- c(basehaz, rep(-999, (m - len)))
ordtreat <- cbind(ordtreat, martfull, hazfull, basefull)
names(ordtreat)[14:16] <- c("mart", "hazard", "basehaz")
ind <- order(ordtreat$time, rev(ordtreat$death), ordtreat$treat)
orddata <- ordtreat[ind, ]
names(orddata) <- names(ordtreat)
p <- 1 #points to the index of the first basehaz != -999
while(orddata$basehaz[p]==-999) p <- p+1

# replaces all treatment values with the last known (placebo) estimate
for(i in (p+1):m)
{
  if(orddata$basehaz[i] == -999)
    orddata$basehaz[i] <- orddata$basehaz[i - 1]
  if(orddata$hazard[i] == -999)
    orddata$hazard[i] <- orddata$basehaz[i] * exp(beta[1] * orddata$x1[i] + beta[2] *
    orddata$x3[i] + beta[3] * orddata$x1[i] * orddata$x3[i] + coeffmax *
    orddata$ind2x2[i] * orddata$x5[i] * orddata$treat[i] + coeffmin * orddata$x4[i]
    * orddata$x5[i] * orddata$x6[i] * orddata$treat[i])
  if(orddata$mart[i] == -999)
    orddata$mart[i] <- orddata$death[i] - orddata$hazard[i]
}

# errases the first treatment values which cannot be estimated with placebo values
orddata <- orddata[-(1:(p-1)),]
orddata$n <- c(1:dim(orddata)[1])
return(orddata)
}

".model.fit.devi"<-function(daten, coeffmin = 1, coeffmax = -2)
{
  ind <- order(daten$treat, daten$time, rev(daten$death))
  ordtreat <- daten[ind, ]
  names(ordtreat) <- names(daten)
  # omits all (first) rows in the data set for which delta(t1)=0
  while(ordtreat$death[1] == 0) ordtreat <- ordtreat[-1, ]
  treat <- ordtreat[, 9]
  plac <- ordtreat[treat == 0, ]
  coxmod <- coxph(Surv(time, death) ~ x1 + x3 + x1 * x3, data = plac, method = "breslow", x
= T)
  coxdetail <- coxph.detail(coxmod)
  beta <- as.vector(coxmod$coefficients)

```

```

mart <- coxmod$residuals
hazard <- plac$death - mart
plac.linpred <- beta[1] * plac$x1 + beta[2] * plac$x3 + beta[3] * plac$x1 * plac$x3
basehaz <- hazard/exp(plac.linpred)
len <- length(mart)
m <- dim(ordtreat)[1]
martfull <- c(mart, rep(-999, (m - len)))
hazfull <- c(hazard, rep(-999, (m - len)))
basefull <- c(basehaz, rep(-999, (m - len)))
ordtreat <- cbind(ordtreat, martfull, hazfull, basefull)
names(ordtreat)[14:16] <- c("mart", "hazard", "basehaz")
ind <- order(ordtreat$time, rev(ordtreat$death), ordtreat$treat)
orddata <- ordtreat[ind, ]
names(orddata) <- names(ordtreat)
p <- 1 #points to the index of the first basehaz != -999
while(orddata$basehaz[p] == -999) p <- p + 1
# replaces all treatment values with the last known (placebo) estimate
for(i in (p + 1):m) {
  if(orddata$basehaz[i] == -999)
    orddata$basehaz[i] <- orddata$basehaz[i - 1]
  if(orddata$hazard[i] == -999)
    orddata$hazard[i] <- orddata$basehaz[i] * exp(beta[1] * orddata$x1[i] + beta[2] *
    orddata$x3[i] + beta[3] * orddata$x1[i] * orddata$x3[i] + coeffmax *
    orddata$ind2x2[i] * orddata$x5[i] * orddata$treat[i] + coeffmin * orddata$x4[i]
    * orddata$x5[i] * orddata$x6[i] * orddata$treat[i])
  if(orddata$mart[i] == -999)
    orddata$mart[i] <- orddata$death[i] - orddata$hazard[i]
}
# errases the first treatment values which cannot be estimated with placebo values
orddata <- orddata[ - (1:(p - 1)), ]
orddata$n <- c(1:dim(orddata)[1]) # calculates deviance residuals
devi <- rep(0, dim(orddata)[1])
for(i in 1:dim(orddata)[1])
  if(orddata$mart[i] >= 0)
    devi[i] <- sqrt(-2 * (orddata$mart[i] + orddata$death[i] * log(orddata$death[i] -
    orddata$mart[i]))) else devi[i] <- - sqrt(-2 * (orddata$mart[i] + orddata$death[i]
    * log(orddata$death[i] - orddata$mart[i])))
orddata <- cbind(orddata, devi)
names(orddata)[17] <- "devi"
return(orddata)
}

".restrict" <- function(prepdata, maxrow)
{
  vari <- maxrow
  if(vari == "x10")
  {

```

```
x <- prepdata[,2]
prepdata <- prepdata[x != 0,]
}
if(vari == "x11")
{
  x <- prepdata[,2]
  prepdata <- prepdata[x != 1,]
}
if(vari == "x20")
{
  x <- prepdata[,3]
  prepdata <- prepdata[x != 0,]
}
if(vari == "x21")
{
  x <- prepdata[,3]
  prepdata <- prepdata[x != 1,]
}
if(vari == "x22")
{
  x <- prepdata[,3]
  prepdata <- prepdata[x != 2,]
}
if(vari == "x30")
{
  x <- prepdata[,4]
  prepdata <- prepdata[x != 0,]
}
if(vari == "x31")
{
  x <- prepdata[,4]
  prepdata <- prepdata[x != 1,]
}
if(vari == "x40")
{
  x <- prepdata[,5]
  prepdata <- prepdata[x != 0,]
}
if(vari == "x41")
{
  x <- prepdata[,5]
  prepdata <- prepdata[x != 1,]
}
if(vari == "x50")
{
  x <- prepdata[,6]
  prepdata <- prepdata[x != 0,]
```

```

    }
    if(vari == "x51")
    {
        x <- prepdata[,6]
        prepdata <- prepdata[x != 1,]
    }
    if(vari == "x60")
    {
        x <- prepdata[,7]
        prepdata <- prepdata[x != 0,]
    }
    if(vari == "x61")
    {
        x <- prepdata[,7]
        prepdata <- prepdata[x != 1,]
    }
    return(prepdata)
}

```

2. Functions needed for simulations with regression trees (on Windows)

```

".100.trees" <- function(nsim = 1, coeffmin = 2, coeffmax = -2, cens.max = 11)
{
    min.percent.correct <- rep(0, nsim)
    max.percent.correct <- rep(0, nsim)
    for(k in 1:nsim) {
        result <- .sim.tree(coeffmin = coeffmin, coeffmax = coeffmax, cens.max =
cens.max)
        min.percent.correct[k] <- result[[1]]
        max.percent.correct[k] <- result[[2]]
    }
    return(min.percent.correct, max.percent.correct)
}

".100.trees.devi" <- function(nsim = 1, coeffmin = 2, coeffmax = -2, cens.max = 11)
{
    min.percent.correct <- rep(0, nsim)
    max.percent.correct <- rep(0, nsim)
    for(k in 1:nsim) {
        result <- .sim.tree.devi(coeffmin = coeffmin, coeffmax = coeffmax, cens.max =
cens.max)
        min.percent.correct[k] <- result[[1]]
        max.percent.correct[k] <- result[[2]]
    }
}

```

```

    mean.min <- mean(min.percent.correct)
    mean.max <- mean(max.percent.correct)
    return(min.percent.correct, max.percent.correct, mean.min, mean.max)
}

".plac.prep" <- function(daten)
{
  treat <- daten[, 9]
  tempdata <- daten[treat == 0, ]
  names(tempdata) <- names(daten)
  placdata <- tempdata[, c(14, 2, 3, 5:9, 12, 13)]
  m <- mean(placdata[, 4])
  for(i in 1:dim(placdata)[1])
    if(placdata[i, 4] <= m) placdata[i, 4] <- 0 else placdata[i, 4] <- 1
  return(placdata)
}

".plac.prep.devi" <- function(daten)
{
  treat <- daten[, 9]
  tempdata <- daten[treat == 0, ]
  names(tempdata) <- names(daten)
  placdata <- tempdata[, c(17, 2, 3, 5:9, 12, 13)]
  m <- mean(placdata[, 4])
  for(i in 1:dim(placdata)[1])
    if(placdata[i, 4] <= m) placdata[i, 4] <- 0 else placdata[i, 4] <- 1
  return(placdata)
}

".sim.tree" <- function(coeffmin = 2, coeffmax = -2, cens.max = 11)
{
  simdata <- .data.sim(coeffmin, coeffmax, cens.max)
  martdata <- .model.fit(simdata, coeffmin, coeffmax)
  placdata <- .plac.prep(martdata)
  treatdata <- .treat.prep(martdata)
  boxtree <- tree(mart ~ x1 + x2 + x3 + x4 + x5 + x6, data = treatdata)
  treattree <- prune.tree(boxtree, best = 5)
  endnodes <- treattree$frame
  ind <- order(endnodes$var, endnodes$yval)
  ordnodes <- endnodes[ind, ]
  var <- ordnodes[, 1]
  ordnodes <- ordnodes[var == "<leaf", ]
  ordnodes <- ordnodes[, -3]
  ordnodes <- ordnodes[, -4]
  ordnodes <- ordnodes[, -4]
  ordnodes <- ordnodes[, -1]
  pos.ind <- rep(0, dim(ordnodes)[1])

```

```

for(i in 1:dim(ordnodes)[1])
  if(ordnodes$yval[i] >= 0) pos.ind[i] <- 1
ordnodes <- cbind(ordnodes, pos.ind)
names(ordnodes)[3] <- "pos.ind"
ind.treat <- identify.tree(treattree, nodes = row.names(ordnodes))
plactree <- predict.tree(treattree, newdata = placdata, type = "tree")
ind.plac <- identify.tree(plactree, nodes = row.names(ordnodes))
n <- dim(ordnodes)[1]
lr <- rep(-1, n)
p <- 0
if(ordnodes[p + 1, 3] != 1) {
  plactemp <- placdata[ind.plac[[1]], ]
  treattemp <- treatdata[ind.treat[[1]], ]
  mincurrent <- rbind(plactemp, treattemp)
  lr[1] <- .logrank(S = Surv(mincurrent$time, mincurrent$death), group =
mincurrent$treat)
  minregion <- mincurrent
  p <- 1
  while(ordnodes[p + 1, 3] != 1 && p + 1 <= n) {
    p <- p + 1
    plactemp <- placdata[ind.plac[[p]], ]
    treattemp <- treatdata[ind.treat[[p]], ]
    mincurrent <- rbind(plactemp, treattemp)
    minregion <- rbind(minregion, mincurrent)
    lr[p] <- .logrank(S = Surv(minregion$time, minregion$death), group =
minregion$treat)
  }
}
if(ordnodes[n, 3] == 1) {
  plactemp <- placdata[ind.plac[[n]], ]
  treattemp <- treatdata[ind.treat[[n]], ]
  maxcurrent <- rbind(plactemp, treattemp)
  lr[n] <- .logrank(S = Surv(maxcurrent$time, maxcurrent$death), group =
maxcurrent$treat)
  maxregion <- maxcurrent
  q <- 1
  while(ordnodes[n - q, 3] == 1 && n - q > p) {
    plactemp <- placdata[ind.plac[[n - q]], ]
    treattemp <- treatdata[ind.treat[[n - q]], ]
    maxcurrent <- rbind(plactemp, treattemp)
    maxregion <- rbind(maxregion, maxcurrent)
    lr[n - q] <- .logrank(S = Surv(maxregion$time, maxregion$death), group =
maxregion$treat)
    q <- q + 1
  }
}
ind <- 1

```

```

while(ind + 1 <= p && lr[ind] > lr[ind + 1]) ind <- ind + 1
indm <- n
while(indm - 1 > p && lr[indm] > lr[indm - 1]) indm <- indm - 1
tree <- rep(0, dim(treatdata)[1])
treatdata <- cbind(treatdata, tree, tree)
names(treatdata)[11] <- "tree"
names(treatdata)[12] <- "actual"
treatdata[ind.treat[[ind]], 11] <- -1
treatdata[ind.treat[[indm]], 11] <- 1
for(i in 1:dim(treatdata)[1]) {
  if(treatdata[i, 3] == 2 && treatdata[i, 6] == 1)
    treatdata[i, 12] <- 1
  if(treatdata[i, 5] == 1 && treatdata[i, 6] == 1 && treatdata[i, 7] == 1 && treatdata[i,
12] == 0)
    treatdata[i, 12] <- -1
}
tabelle <- table(treatdata$tree, treatdata$actual)
sums <- colSums(tabelle)
min.percent.correct <- (100 * tabelle[1, 1])/sums[1]
max.percent.correct <- (100 * tabelle[3, 3])/sums[3]
return(min.percent.correct, max.percent.correct)
}

".sim.tree.devi" <- function(coeffmin = 2, coeffmax = -2, cens.max = 11)
{
  simdata <- .data.sim(coeffmin, coeffmax, cens.max)
  martdata <- .model.fit.devi(simdata, coeffmin, coeffmax)
  placdata <- .plac.prep.devi(martdata)
  treatdata <- .treat.prep.devi(martdata)
  boxtree <- tree(devi ~ x1 + x2 + x3 + x4 + x5 + x6, data = treatdata)
  treattree <- prune.tree(boxtree, best = 5)
  endnodes <- treattree$frame
  ind <- order(endnodes$var, endnodes$yval)
  ordnodes <- endnodes[ind, ]
  var <- ordnodes[, 1]
  ordnodes <- ordnodes[var == "<leaf>", ]
  ordnodes <- ordnodes[, -3]
  ordnodes <- ordnodes[, -4]
  ordnodes <- ordnodes[, -4]
  ordnodes <- ordnodes[, -1]
  pos.ind <- rep(0, dim(ordnodes)[1])
  for(i in 1:dim(ordnodes)[1])
    if(ordnodes$yval[i] >= 0) pos.ind[i] <- 1
  ordnodes <- cbind(ordnodes, pos.ind)
  names(ordnodes)[3] <- "pos.ind"
  ind.treat <- identify.tree(treattree, nodes = row.names(ordnodes))
  plactree <- predict.tree(treattree, newdata = placdata, type = "tree")
}

```

```

ind.plac <- identify.tree(plactree, nodes = row.names(ordnodes))
n <- dim(ordnodes)[1]
lr <- rep(-1, n)
p <- 0
if(ordnodes[p + 1, 3] != 1) {
  plactemp <- placdata[ind.plac[[1]], ]
  treattemp <- treatdata[ind.treat[[1]], ]
  mincurrent <- rbind(plactemp, treattemp)
  lr[1] <- .logrank(S = Surv(mincurrent$time, mincurrent$death), group =
mincurrent$treat)
  minregion <- mincurrent
  p <- 1
  while(ordnodes[p + 1, 3] != 1 && p + 1 <= n) {
    p <- p + 1
    plactemp <- placdata[ind.plac[[p]], ]
    treattemp <- treatdata[ind.treat[[p]], ]
    mincurrent <- rbind(plactemp, treattemp)
    minregion <- rbind(minregion, mincurrent)
    lr[p] <- .logrank(S = Surv(minregion$time, minregion$death), group =
minregion$treat)
  }
}
if(ordnodes[n, 3] == 1) {
  plactemp <- placdata[ind.plac[[n]], ]
  treattemp <- treatdata[ind.treat[[n]], ]
  maxcurrent <- rbind(plactemp, treattemp)
  lr[n] <- .logrank(S = Surv(maxcurrent$time, maxcurrent$death), group =
maxcurrent$treat)
  maxregion <- maxcurrent
  q <- 1
  while(ordnodes[n - q, 3] == 1 && n - q > p) {
    plactemp <- placdata[ind.plac[[n - q]], ]
    treattemp <- treatdata[ind.treat[[n - q]], ]
    maxcurrent <- rbind(plactemp, treattemp)
    maxregion <- rbind(maxregion, maxcurrent)
    lr[n - q] <- .logrank(S = Surv(maxregion$time, maxregion$death), group =
maxregion$treat)
    q <- q + 1
  }
}
ind <- 1
while(ind + 1 <= p && lr[ind] > lr[ind + 1]) ind <- ind + 1
indm <- n
while(indm - 1 > p && lr[indm] > lr[indm - 1]) indm <- indm - 1
tree <- rep(0, dim(treatdata)[1])
treatdata <- cbind(treatdata, tree, tree)
names(treatdata)[11] <- "tree"

```

```

names(treatdata)[12] <- "actual"
treatdata[ind.treat[[ind]], 11] <- -1
treatdata[ind.treat[[indm]], 11] <- 1
for(i in 1:dim(treatdata)[1]) {
  if(treatdata[i, 3] == 2 && treatdata[i, 6] == 1)
    treatdata[i, 12] <- 1
  if(treatdata[i, 5] == 1 && treatdata[i, 6] == 1 && treatdata[i,7] == 1 && treatdata[i,
12] == 0)
    treatdata[i, 12] <- -1
}
tabelle <- table(treatdata$tree, treatdata$actual)
sums <- colSums(tabelle)
min.percent.correct <- (100 * tabelle[1, 1])/sums[1]
max.percent.correct <- (100 * tabelle[3, 3])/sums[3]
return(min.percent.correct, max.percent.correct)
}

".treat.prep" <- function(daten)
{
  treat <- daten[, 9]
  tempdata <- daten[treat == 1, ]
  names(tempdata) <- names(daten)
  treatdata <- tempdata[, c(14, 2, 3, 5:9, 12, 13)]
  m <- mean(treatdata[, 4])
  for(i in 1:dim(treatdata)[1])
    if(treatdata[i, 4] <= m) treatdata[i, 4] <- 0 else treatdata[i,4] <- 1
  return(treatdata)
}

".treat.prep.devi" <- function(daten)
{
  treat <- daten[, 9]
  tempdata <- daten[treat == 1, ]
  names(tempdata) <- names(daten)
  treatdata <- tempdata[, c(17, 2, 3, 5:9, 12, 13)]
  m <- mean(treatdata[, 4])
  for(i in 1:dim(treatdata)[1])
    if(treatdata[i, 4] <= m) treatdata[i, 4] <- 0 else treatdata[i,4] <- 1
  return(treatdata)
}

```

3. Functions needed for analysis of EMIAT with stabilized bump hunting (on UNIX)

```

".bumping.borders"<-function(method = 3, train.data, valid.data = 0, varno = 1, type =
      rep(0,dim(train.data)[2] - 1), maxi = T, pasting = T, missing = -9999,
      beta = 1/dim(train.data)[1], alpha = seq(from = 0.05, to = 0.1, by =
      0.005), thinning = T, peel.crit = 2, globalmean = T, legende = T,
      interactive = F, n.samples = 10, train = F, xl = 0, xu = 1, yl =
      min(train.data[train.data[, varno] != missing, varno]), yu =
      max(train.data[train.data[, varno] != missing, varno]), denom = 10,
      lineplot = T, both = T, language = 1)
{
  traj <- .multiple.traj(method = method, train.data = train.data, valid.data = train.data,
      type = type, maxi = maxi, alpha = alpha, beta = beta, peel.crit = peel.crit,
      interactive = interactive, n.samples = n.samples)
  m <- order(traj[[1]][, 6], traj[[1]][, 3])

  # gives a vector of coefficients ordered first by sample no., then by train.ymean as in traj
  traj.sorted <- cbind(traj[[1]][, 1][m], traj[[1]][, 3][m], traj[[1]][, 6][m])
  # contains the columns: beta before pasting, ymean, and sample no.
  traj.sorted <- data.frame(traj.sorted)
  end <- dim(traj[[2]])[2]
  cat("Original Data", "\n")
  for(i in 1:(n.samples + 1)) {
    index.samples <- traj[[2]][, 2:end]
  # extracts the indexes of all bootstrap samples
    data <- train.data[index.samples[i, ], ]
  # takes only the current boot sample data
    beta <- traj.sorted[2, 1]
  # takes the beta before pasting for the current sample
    if(beta > 0 && beta < 1) {
      box <- .boxes(train.data = data, alpha = alpha, beta = beta, nboxes = 1, output = F, maxi
          = maxi, type= type, peel.crit = peel.crit)
      outbox <- .express.boxes(box)
      cat(outbox[[2]], "\n", outbox[[5]], "\n")
    }
    cat("bootstrap sample = ", (i + 1), "\n")
    Sample <- traj.sorted[, 3]
    traj.sorted <- traj.sorted[Sample != i, ]
  }
  return(0)
}

```

```

".bumping.borders.min"<-function(method = 3, train.data, valid.data = 0, varno = 1, type =
      rep(0, dim(train.data)[2] - 1), maxi = F, pasting = T, missing =
      -9999, beta = 1/dim(train.data)[1], alpha = seq(from = 0.05,
      to = 0.1, by = 0.005), thinning = T, peel.crit = 2, globalmean =
      T, legende = T, interactive = F, n.samples = 10, train = F, xl =
      0, xu = 1, yl = min(train.data[train.data[, varno] != missing,
      varno]), yu = max(train.data[train.data[, varno] != missing,
      varno]), denom = 10, lineplot = T, both = T, language = 1)
{
  traj <- .multiple.traj(method = method, train.data = train.data, valid.data = train.data,
      type = type, maxi = maxi, alpha = alpha, beta = beta, peel.crit = peel.crit, ^
      interactive = interactive, n.samples = n.samples)
  m <- order(traj[[1]][, 6], - traj[[1]][, 3])
  # gives a vector of coefficients ordered first by sample no., then by train.ymean as in traj
  traj.sorted <- cbind(traj[[1]][, 1][m], traj[[1]][, 3][m], traj[[1]][, 6][m])
  # contains the columns: beta before pasting, ymean, and sample no.
  traj.sorted <- data.frame(traj.sorted)
  resultat <- traj.sorted
  end <- dim(traj[[2]])[2]
  cat("Original Data", "\n")
  for(i in 1:(n.samples + 1)) {
    index.samples <- traj[[2]][, 2:end]
    # extracts the indexes of all bootstrap samples
    data <- train.data[index.samples[i, ], ]
    # takes only the current boot sample data
    beta <- traj.sorted[2, 1]
    # takes the beta before pasting for the current sample
    if(beta > 0 && beta < 1) {
      box <- .boxes(train.data = data, alpha = alpha, beta = beta, nboxes = 1, output = F, maxi
          = F, type = type, peel.crit = peel.crit)
      outbox <- .express.boxes(box)
      cat(outbox[[2]], "\n", outbox[[5]], "\n")
    }
    cat("bootstrap sample = ", (i + 1), "\n")
    Sample <- traj.sorted[, 3]
    traj.sorted <- traj.sorted[Sample != i, ]
  }
  return(resultat)
}

```

BIBLIOGRAPHY

- Aalen O. *Nonparametric inference in connection with multiple decrement models*. Scandinavian Journal of Statistics 3 (1976): 15–27.
- Aalen, O. *Nonparametric inference for a family of counting processes*. The Annals of Statistics 6 (1978): 701–726.
- Altman, D.G., Lansen B., Sauerbrei, W., Schumacher, M. *Danger of using “optimal” cutpoints in the evaluation of prognostic factors*. Journal of the National Cancer Institute 86.11, 1. June 1994: 829-835.
- Andersen, P.K., Borgan, D., Gill, R.D., Keiding, N. *Statistical Models Based on Counting Processes*. Springer Series in Statistics. New York: Springer Verlag, 1993.
- Barlow, W.E. and Prentice, R.L. *Residuals for relative risk regression*. Biometrika 75 (1988): 65-74.
- Becker, U. *Bump hunting software*. Computer software. Institute for Statistics, LMU Munich, 1999. Unix, downloadable at www.stat.uni-muenchen.de/~ursula/.
- Breiman, L. *Bagging Predictors*. Machine Learning 26 (1996): 123–140.
- Breiman, L., Friedman, J.H., Olsen, R.A., Stone, C.J. *Classification & Regression Trees*. Pacific Grove, CA: Wodsworth & Brooks/Cole, advanced books & software, 1984.
- Breslow, N. E. Covariance analysis of censored survival data. Biometrics, 30 (1974): 89-99.
- Cox, D. R. *Regression models and life-tables (with discussion)*. Journal of the Royal Statistical Society B, 34 (1972): 187-220.

-
- Cox, D. R. & Oakes, D. *Analysis of Survival Data*. Monographs on Statistics and Applied Probability 21. London: Chapman & Hall, 1984.
- Cox, D. R. & Snell, E. J. A general definition of residuals (with discussion). *Journal of the Royal Statistical Society B*, 30 (1968): 248-275.
- Crowley, J. & Hu, M. *Covariance analysis of heart transplant survival data*. *Journal of the American Statistical Association* 72 (1977): 27–36.
- Dannegger F. *Tree stability diagnostics and some remedies for instability*. *Statistics in Medicine* 19 (2000): 475–491.
- Friedman, J.H. & Fisher, N.I. *Bump hunting in high-dimensional data*. *Statistics and Computing* 9.2 (1999): 123–143.
- Gill, R.D. *Understanding Cox's regression model: A Martingale approach*. *Journal of the American Statistical Association* 79 (1984): 441-447.
- Harrell, F.E. Jr. *Regression Modeling Strategies with Applications to Linear Models, Logistic Regression, and Survival Analysis*. Springer Statistical Series, Eds. Bickel, P. et al. New York: Springer Verlag, 2001.
- Harrell, F.E., Lee, K.L., Mark, D.B. *Tutorial in Biostatistics. Multivariable prognostic models: issues in developing models, evaluating assumptions and adequacy, and measuring and reducing errors*. *Statistics in Medicine* 15 (1996): 361–387.
- Hastie, T., Tibshirani, R, Friedman, J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer Verlag, 2001.
- Janse, M.J. et al. *Identification of post acute myocardial infarction patients with potential benefit from prophylactic treatment with Amiodarone. A substudy of EMIAT*. *European Heart Journal* 19 (1998): 85–95.
- Julian, D.G. et al. *Randomised trial of effect of Amiodarone on mortality in patients with left-ventricular dysfunction after recent myocardial infarction: EMIAT*.

-
- The Lancet 349, 8. March 1997: 667-674.
- Kay, R. *Proportional hazards regression models and the analysis of censored survival data*. Applied Statistics 26 (1977): 227–237.
- Malik, M. et al. *Depressed Heart Rate Variability Identifies Postinfarction Patients who Might Benefit From Prophylactic Treatment with Amiodarone. A Substudy of EMIAT*. Journal of the American College of Cardiology 35.5, April 2000: 1263-1275.
- Marubini, E. & Valsecchi, M. G. *Analysing Survival Data from Clinical Trials and Observational Studies*. Statistics in Practice. Ed. Barnett, V. Chichester: John Wiley & Sons, 1995.
- Nardi, A. and Schemper, M. *New Residuals for Cox Regression and Their Application to Outlier Screening*. Biometrics 55, June 1999: 523-529.
- Peto, R. *Discussion of Professor Cox's paper*. Journal of the Royal Statistical Society B, 34 (1972): 205-207.
- Schemper, M. *Non-parametric analysis of treatment-covariate interaction in the presence of censoring*. Statistics in Medicine 7 (1998): 1257–1266.
- Schoenfeld, D. *Partial Residuals for the proportional hazards regression model*. Biometrika 69 (1982): 239-241.
- Silva, O.E. & Zurrida, S. *Breast Cancer: A Practical Guide*. New York: Elsevier, 2000.
- Therneau, T.M., Grambsch, P.M. *Modeling Survival Data. Extending the Cox Model*. Statistics for Biology and Health. New York: Springer Verlag, 2000.
- Therneau, T.M., Grambsch, P.M., and Fleming, T.R. *Martingale hazards regression models and the analysis of censored survival data*. Biometrika 77 (1990): 147-160.

Therneau, T.M., Grambsch, P.M., Fleming, T.R. *Martingale-based residuals for survival models*. Biometrika 77.1 (1990), 147–160.

Tibshirani, R. & Knight, K. *Model Search by Bootstrap “Bumping”*. Journal of Computational & Graphical Statistics 8.4 , December 1999.

ZUSAMMENFASSUNG

Klinische Untersuchungen beurteilen die Wirksamkeit einer neuen Behandlungsmethode oftmals dadurch, dass sie Patienten zufällig der neuen oder einer etablierten Behandlungsmethode zuweisen und danach die Ergebnisse (Überlebensrate) vergleichen. Gewöhnlicherweise wird dabei die komplette Patientengruppe analysiert obwohl bekannt ist, dass bestimmte Untergruppen unterschiedlich auf die neue Behandlungsmethode reagieren. Einige der Patienten profitieren von der neuen Behandlung (positive Respondern), während andere dadurch zu Schaden kommen (negative Respondern). Das Ziel dieser Dissertation ist es, solche Untergruppen von Patienten zu identifizieren. Erreicht wird es dadurch dass man sogenannte prädiktive Faktoren findet, die unterschiedliche Überlebenswahrscheinlichkeiten nur anhand von Unterschieden in der Behandlungsmethode beschreiben.

Diese Dissertation beginnt mit einer Übersicht über Techniken, die bisher zur Responderidentifikation benutzt wurden und schlägt gleichzeitig eine neue Methode zur systematischen Suche nach Respondern vor. Diese neue Methode besteht aus den folgenden drei Schritten:

1. Identifikation von prognostischen Faktoren hinsichtlich der neuen Behandlung (Zum Beispiel durch das Cox-PH Model angewandt auf die Teilgruppe der Patienten die man der Standardbehandlung unterzieht). Diese Faktoren sind 'prognostisch' im klassischen Sinne für den Fall, dass in der Studie gegen Placebo und nicht eine Standardmethode getestet wurde.
2. Identifizierung der Patientengruppe(n) aus der Neubehandlungsgruppe, deren Überlebenswahrscheinlichkeiten nur schlecht durch das Prognosemodel vorhergesagt wurden. (Zum Beispiel durch eine Suche nach Ausreißern in der Devianz- oder Martingaleresiduen.)
3. Identifikation der prädiktive Faktoren, welche die gemeinsamen Eigenschaften von Patienten mit Residuenausreißern beschreiben (positive und negative Respondern).

Dies führt man mit Regressionsbäumen, bump hunting oder mit dem vorgeschlagenen stabilisierten bump hunting durch.

Die Methode zur Identifizierung von Respondern wurde zur Analyse von Daten klinischer Studien entwickelt, in denen kein Unterschied in den Überlebenswahrscheinlichkeiten von Patienten, die nach der alten oder der neuen Behandlungsmethode versorgt wurden, festzustellen ist. Änderungen an der Methode für den Fall, dass bei den Daten der beiden Patientengruppen doch Unterschiede in den Überlebenskurven bestehen, wurden diskutiert. Darüber hinaus wurden Variationen der Responderidentifikation vorgeschlagen und in einer Simulationsstudie verglichen.

Bei der Suche nach prädiktive Faktoren kann man Martingale- oder Devianzresiduen auf das Prognosemodell als Responsevariable im Regressionsbaum, bump hunting oder stabilisierten bump hunting Prozess anwenden. Die Simulationsstudie hat gezeigt, dass Martingaleresiduen kombiniert mit dem stabilisierten bump hunting Prozess am geeignetsten für die Identifizierung von Respondern ist. Diese Variante des vorgeschlagenen Prozesses identifizierte positive und negative Respondern in 99% aller Fälle.

Einige Variationen der vorgeschlagenen Methode zur Identifikation von Respondern wurden auch auf einen ‚echten‘ Datensatz (dem European Myocardial Infarction Amiodarone Trial EMIAT) angewandt. Die dabei identifizierten Gruppen von positiven und negativen Respondern wurden verglichen.

Alle Variationen des beschriebenen Algorithmus zur Identifikation von Respondern und speziell der Prozess mit stabilisiertem bump hunting mit martingale Residuen als Response zeigen eine bessere Leistung als die momentan verwendete Cox-PH Methode mit Behandlungsinteraktionen. Dies wurde anhand der Simulationsstudie und anhand der Daten von EMIAT gezeigt. Die besseren Ergebnisse der neuen Methode erklären sich aus der Tatsache, dass es im Vergleich zu dem Cox-PH Model viel leichter Interaktionen höherer Ordnung zwischen Kovariablen erkennt.

Ausblick

Eine vollständige Implementierung der sechs verschiedenen Variationen des Algorithmus zur Responderidentifizierung mit festgelegter Art und Anzahl von Faktoren wurde für die Simulationsstudie durchgeführt. Die Responderanalyse der EMIAT-Daten wurde teilautomatisch durchgeführt. Dadurch erhält man einerseits mehr Flexibilität bei der Bestimmung eines prädiktiven Modells, verlangsamt andererseits aber die Analyse der Daten. Für eine zukünftige Verwendung des nachweislich besten Algorithmus, stabilisiertes bump hunting mit Martingaleresiduen, macht es Sinn, eine vollständige Implementierung mit frei wählbaren Einstellungen für die Anzahl und Art der Faktoren, verschiedenen Größen von Datensätzen, und der Stopkriterien für den Bau des Modells zu verwirklichen.

Die Methode zur Identifizierung von Respondern wurde für Datensätze entwickelt, bei denen die beiden Patientengruppen (klassische und neue Behandlungsmethode) keine Unterschiede in den Überlebenskurven zeigen, und auch an diesen getestet. Ein zukünftiges Softwareprodukt zur Responderanalyse sollte dieser Beschränkung nicht mehr unterliegen und auch Datensätze sinnvoll analysieren können, bei denen bei den beiden Patientengruppen Unterschiede in der Überlebenscharakteristik bestehen. Die Leistung eines solchen Programms könnte man dann auch wieder anhand simulierte Daten testen.

Bisher wurde angenommen, dass weder prognostische noch prädiktive Faktoren sich mit der Zeit ändern. Eine neue Studie könnte durchgeführt, werden die Zeitabhängigkeiten bei der Identifikation von Respondern berücksichtigt.

Lebenslauf

Victoria Ivanova Kehl

Gymnasium: **Gymnasium der Wissenschaft und Mathematik „Acad. N. Obreshkov“**
Burgas, Bulgarien

Universitäten:

1993-97	Southern Oregon University , Ashland, Oregon, U. S. A. Studium der Mathematik mit Nebenfach Informatik
Juni 1997	<u>Abschluß:</u> <i>Bachelor's Degree in Mathematics</i> , mit Auszeichnung
1997-99	Clemson University , South Carolina, U. S. A. Studium der Mathematik mit Schwerpunkt Statistik
August 1999	<u>Abschluß:</u> <i>Master in Mathematical Sciences</i>
1998-99	Universität Kaiserslautern Teilnahme am Programm „Mathematics International“
September 1999	<u>Abschluß:</u> <i>Master bei Mathematics International</i>
2000-2002	Ludwig-Maximilians-Universität , München Promotionsstudium in Statistik
November 2002 (erwartet)	<u>Abschluß:</u> <i>Promotion, Dr. rer. nat.</i>

Abschluß Arbeiten:

<i>Bachelor's Arbeit</i>	„Mathematisches modellieren des HIV-Virus in Interaktion mit dem menschlichen Immunsystem.“
<i>Master Arbeit</i>	„Beeinflussung des Gewichtszuwachses während der Schwangerschaft von Minderjährigen durch psycho-soziale Faktoren, und deren beider Einfluß auf Geburtsgewicht von Babies.“
<i>Dissertation</i>	„Responder identification in Clinical Trials“

Lehrerfahrung und Wissenschaftliche Arbeit:

1997	Assistentenstelle an der Southern Oregon University <ul style="list-style-type: none">• Übungsgruppenleiter in Analysis
1997-98	Lehrassistentenstelle an der Clemson University <ul style="list-style-type: none">• Statistik• Analysis
1998-99	Wissenschaftliche Assistentin an dem Institut für Techno-und-Wirtschaftsmathematik, Kaiserslautern <ul style="list-style-type: none">• Statistische Analyse von Daten des Öffentlichen Verkehrs
Seit November 1999	Wissenschaftliche Mitarbeiterin an dem Institut für Medizinische Statistik und Epidemiologie, TU – München <ul style="list-style-type: none">• Überlebensanalyse• Responderanalyse

Gesellschaftliches Engagement:

Während meiner Zeit an der Southern Oregon University, war ich „International Cultural Service Program Coordinator“. Des weiteren war ich im Studentenparlament („Inter-club Council Member“) und der Studentenverwaltung als Abgeordnete tätig.

Auszeichnungen und Ehrungen:

Lithia Springs Rotary Club Scholarship

1994-95

International Student Fee Remission – Studiengebühren Übernahme vom Staat Oregon

1993-95

International Cultural Service Program – Studiengebühren Übernahme vom Staat Oregon für ausländische Studenten im Austausch für kulturelle gemeinnützige Arbeit.

1993-95 – Teilnehmer am Programm

1995-97 – Hilfs-Organisator des Programmes

Churchill Scholars Honors Program – Oregon Laurels Scholarship – wird an Studenten des „Ethics Studies Program“ vergeben.

1994-95

Ida and Eugene Bowman Scholarship – wird an herausragende Studenten der Mathematik vergeben.

1995-96

Harry S. Keival Auszeichnungen – wird an herausragende Studenten der Mathematik vergeben.

1995-96

Internationaler Mathematik Wettbewerb in Modellieren - Auszeichnung: „Honorable Mention“

1997

Universidad de Guanajuato's International Student Achievement Award at Southern Oregon

University – Auszeichnung für exzellente akademische Arbeit, Universitäts politisches Engagement, und Beitrag zum internationalen Verständnis.

1997

Vorgeschlagen für „Associate Member of ΣX – Scientific research Society“

1997

American Association of University Women's Outstanding Senior Woman Award

1997

Vorgeschlagen für „Who's Who Among Students in American Universities and Colleges“

1996-97

Bachelor's Abschluß in Mathematik mit Auszeichnung

1997

Sprachen:

Bulgarisch	– Muttersprache
Englisch	– fließend
Deutsch	– fließend
Russisch	– gut

Referenzen:

- Prof. Dr. K. Ulm, Institut für Medizinische Statistik und Epidemiologie
TU – München, Ismaningerstr. 22, D-80675 München
- Dr. H. Senter, Department of Mathematical Sciences,
Clemson University, Martin Hall, Box 341907, Clemson, SC 29634-1907, U. S. A.
- Dr. K. Yates, Department of Mathematics,
Southern Oregon University, Ashland, OR 97520-5026, U. S. A.